REJ10J1655-0100

Everywhere you imagine.

RENESAS

E8a Emulator

User's Manual

Renesas Microcomputer Development Environment System

R0E00008AKCE00E

User's Manual

Rev.1.00
Revision Date: Jun. 21, 2007

Renesas Technology
www.renesas.com

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

Introduction

This manual describes the followings

- E8a Emulator functions
- Preparation before use
- Debugger functions
- Tutorial
- Appendix: (Components of the E8a Emulator, Notes on High-performance Embedded Workshop, and Diagnostic Test Procedure)

This manual does not intend to explain how to write C/C++ or assembly language programs, how to use any particular operating system or how best to tailor code for the individual devices. These issues are left to the respective manuals.

Microsoft® and Windows® are registered trademarks of Microsoft Corporation.

IBM is a registered trademark of International Business Machines Corporation.

All brand or product names used in this manual are trademarks or registered trademarks of their respective companies or organizations.

Document Conventions

This manual uses the following typographic conventions:

Table 1 Typographic Conventions

Convention	Meaning
[Menu->Menu Option]	Bold text with '->' is used to indicate menu options (for example, **[File->Save As...]**).
FILENAME.C	Uppercase names are used to indicate filenames.
"enter this string"	Used to indicate text that must be entered (excluding the "" quotes).
Key + Key	Used to indicate required key presses. For example, **CTRL+N** means press the **CTRL** key and then, whilst holding the **CTRL** key down, press the **N** key.
➲ (The "how to" symbol)	When this symbol is used, it is always located in the left hand margin. It indicates that the text to its immediate right is describing "how to" do something.

IMPORTANT

READ FIRST

Before using this product, be sure to read the user's manual (this user's manual) carefully.
Keep this user's manual, and refer to this when you have questions about this product.

Emulator:

The emulator in this document refers to the following products that are manufactured by Renesas Technology Corp.:
(1) Emulator
(2) User system interface cable
The emulator herein does not include the customer's user system and host machine.

Purpose of use of the emulator:

This emulator is a device to support the development of a system that uses the Renesas MCUs. It provides support for system development in both software and hardware.
Be sure to use this emulator correctly according to said purpose of use. Please avoid using this emulator for other than its intended purpose of use.

For those who use this emulator:

This emulator can only be used by those who have carefully read the user's manual and know how to use it.
Use of this emulator requires the basic knowledge of electric circuits, logical circuits, and MCUs.

When using the emulator:

(1) This product is a development supporting unit for use in your program development and evaluation stages. In mass-producing your program you have finished developing, be sure to make a judgment on your own risk that it can be put to practical use by performing integration test, evaluation, or some experiment else.
(2) In no event shall Renesas Solutions Corp. be liable for any consequence arising from the use of this product.
(3) Renesas Solutions Corp. strives to renovate or provide a workaround for product malfunction at some charge or without charge. However, this does not necessarily mean that Renesas Solutions Corp. guarantees the renovation or the provision under any circumstances.
(4) This product has been developed by assuming its use for program development and evaluation in laboratories. Therefore, it does not fall under the application of Electrical Appliance and Material Safety Law and protection against electromagnetic interference when used in Japan.
(5) Renesas Solutions Corp. cannot predict all possible situations or possible cases of misuse where a potential danger exists. Therefore, the warnings written in this user's manual and the warning labels attached to this emulator do not necessarily cover all of such possible situations or cases. Please be sure to use this emulator correctly and safely on your own responsibility.
(6) This product is not qualified under UL or other safety standards and IEC or other industry standards. This fact must be taken into account when taking this product from Japan to some other country.

Usage restrictions:

This emulator has been developed as a means of supporting system development by users. Therefore, do not use it as a device used for equipment-embedded applications. Also, do not use it for developing the systems or equipment used for the following purposes either:

(1) Transportation and vehicular
(2) Medical (equipment where human life is concerned)
(3) Aerospace
(4) Nuclear power control
(5) Undersea repeater

If you are considering the use of this emulator for one of the above purposes, please be sure to consult your local distributor.

About product changes:

We are constantly making efforts to improve the design and performance of this emulator. Therefore, the specification or design of this emulator or its user's manual may be changed without prior notice.

About the rights:

We assume no responsibility for any damage or infringement on patent rights or any other rights arising from the use of any information, products or circuits presented in this user's manual.

(1) We assume no responsibility for any damage or infringement on patent rights or any other rights arising from the use of any information, products or circuits presented in this user's manual.
(2) The information or data in this user's manual does not implicitly or otherwise grant a license for patent rights or any other rights belonging to us or third parties.
(3) This user's manual and this emulator are copyrighted, with all rights reserved by us. This user's manual may not be copied, duplicated or reproduced, in whole or part, without prior written consent of us.

About diagrams:

The diagrams in this user's manual may not all represent exactly the actual object.

Device names:

Sections 1 through 6 in this user's manual use R8C/Tiny as an example of the device names.

Guarantee:

If your product becomes faulty within one year after its purchase while being used under good conditions by observing "IMPORTANT" and "Precautions for Safety" described in this user's manual, we will repair or replace your faulty product free of charge. Note, however, that if your product's fault is raised by any one of the following causes, we will repair it or replace it with new one with extra-charge:

- Misuse, abuse, or use under extraordinary conditions
- Unauthorized repair, remodeling, maintenance, and so on
- Inadequate user's system or misuse of it
- Fires, earthquakes, and other unexpected disasters

In the above cases, contact your local distributor.

Precautions for Safety

READ FIRST

Before using this product, be sure to read the user’s manual (this user's manual) carefully.

Keep this user’s manual, and refer to this when you have questions about this product.

DEFINITION OF SIGNAL WORDS

Be sure to read and understand the warnings below before using this emulator. Note that these are the main warnings, not the complete list.

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

⚠ DANGER **DANGER** indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury.

⚠ WARNING **WARNING** indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury.

⚠ CAUTION **CAUTION** indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury.

CAUTION **CAUTION** used without the safety alert symbol indicates a potentially hazardous situation which, if not avoided, may result in property damage.

NOTE emphasizes essential information.

⚠ WARNING

1. **Do not repair or remodel the emulator product by yourself for electric shock prevention and quality assurance.**
2. **Always switch OFF the host computer and user system before connecting or disconnecting any CABLES or PARTS.**
   Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.
3. **Connect the connectors in the user system and in the user interface cable by confirming the correct direction.**

CAUTION

Place the host computer and user system so that no cable is bent or twisted. A bent or twisted cable will impose stress on the user interface leading to connection or contact failure. Make sure that the host computer and the user system are placed in a secure position so that they do not move during use nor impose stress on the user interface.

Contents

Section 1 Overview 1
1.1 Warnings 4
1.2 Environmental Conditions 5
1.3 Components 6

Section 2 E8a Emulator Functions 7
2.1 Overview 7
2.2 Trace Functions 9
2.3 Break Functions 10
2.4 Memory Access Functions 11
2.5 Stack Trace Function 12
2.6 Start/Stop Function 12
2.7 Online Help 13

Section 3 Preparation before Use 15
3.1 Emulator Preparation 15
3.2 Emulator Hardware Configuration 16
3.3 Emulator Software Configuration 22
3.3.1 CD 22
3.4 Installing Emulator's Debugger 23
3.5 Connecting the Emulator to the Host Computer 24
3.6 Connecting the Emulator to the User System 26
3.7 Connecting System Ground 28
3.8 System Check 29
3.9 Uninstalling the Emulator's Debugger 40

Section 4 Preparations for Debugging 45
4.1 Method for Activating High-performance Embedded Workshop 45
4.1.1 Creating the New Workspace (Toolchain Not Used) 46
4.1.2 Creating the New Workspace (Toolchain Used) 50
4.1.3 Selecting an Existing Workspace 61
4.2 Setting at E8a Emulator Activation 63
4.2.1 Setting at Emulator Activation 63
4.2.2 Downloading a Program 65
4.2.3 Setting the [Program Flash] Mode 66

Section 5 Debugging 75
5.1 Setting the Environment for Emulation 75
5.1.1 Opening the [Configuration] Dialog Box 75
5.1.2 [General] Page 76
5.2 Downloading a Program 78
5.2.1 Downloading a Program 78
5.2.2 Viewing the Source Code 79
5.3 Stopping Your Program 82

5.3.1 Address Match Breakpoints 82
5.4 Using the Event Points 83
5.4.1 PC Breakpoints 83
5.4.2 Break Conditions 83
5.4.3 Opening the [Event] Window 83
5.4.4 Setting PC Breakpoints 83
5.4.5 Add 84
5.4.6 Edit 84
5.4.7 Enable 84
5.4.8 Disable 84
5.4.9 Delete 84
5.4.10 Delete All 84
5.4.11 Go to Source 85
5.4.12 [Set Break] Dialog Box 85
5.4.13 Setting Break Conditions 86
5.4.14 Edit 87
5.4.15 Enable 87
5.4.16 Disable 88
5.4.17 Delete 88
5.4.18 Delete All 88
5.4.19 Go to Source 88
5.4.20 Sequential Conditions 88
5.4.21 Editing Break Conditions 88
5.4.22 Modifying Break Conditions 88
5.4.23 Enabling Break Conditions 88
5.4.24 Disabling Break Conditions 88
5.4.25 Deleting Break Conditions 89
5.4.26 Deleting All Break Conditions 89
5.4.27 Viewing the Source Line for Break Conditions 89
5.4.28 [Break condition x] Dialog Box 89
5.5 Viewing the Trace Information 91
5.5.1 Opening the [Trace] Window 91
5.5.2 Acquiring Trace Information 91
5.5.3 Clearing the Trace Information 92
5.5.4 Saving the Trace Information in a File 92
5.5.5 Viewing the [Source] Window 92
5.5.6 Trimming the Source 92
5.6 Using the Start/Stop Function 93
5.6.1 Opening the [Start/Stop Function Setting] Dialog Box 93
5.6.2 Specifying the Routine to be Executed 93
5.6.3 Restrictions on the Start/Stop Function 95

Section 6 Tutorial 97
6.1 Introduction 97
6.2 Running the High-performance Embedded Workshop 98
6.3 Setting up the E8a Emulator 98
6.4 Setting the [Configuration] Dialog Box 99
6.5 Checking the Operation of RAM 101
6.6 Downloading the Tutorial Program 103
6.6.1 Downloading the Tutorial Program 103

6.6.2 Displaying the Source Program 104
6.7 Setting a PC Breakpoint 105
6.8 Setting Registers 106
6.9 Executing the Program 108
6.10 Reviewing Breakpoints 111
6.11 Viewing Memory 112
6.12 Watching Variables 113
6.13 Stepping Through a Program 116
6.13.1 Executing [Step In] Command 116
6.13.2 Executing [Step Out] Command 117
6.13.3 Executing [Step Over] Command 118
6.14 Forced Breaking of Program Executions 119
6.15 Displaying Local Variables 120
6.16 Break Function 121
6.16.1 PC Break Function 121
6.17 Hardware Break Function 125
6.18 Trace Functions 128
6.18.1 Displaying the Trace Window 128
6.19 Stack Trace Function 129
6.20 What Next? 131

Appendix A Components of the E8a Emulator 133

Appendix B Window Functions 135

Appendix C Command-Line Functions 139

Appendix D Notes on High-performance Embedded Workshop 141

Appendix E E8aSCP (E8a Self-Check Program) 147
E.1. System Setup to Execute the E8aSCP 147
E.2. Executing the Self-Check Program 150

Section 1 Overview

The High-performance Embedded Workshop is a Graphical User Interface intended to ease the development and debugging of applications written in C/C++ programming language and assembly language for Renesas microcomputers. Its aim is to provide a powerful yet intuitive way of accessing, observing and modifying the debugging platform in which the application is running.

The E8a emulator (hereafter referred to as the emulator) is a software and hardware development support tool for application systems using the Renesas microcomputer. For applicable MCUs, refer to the web site.

The main unit of the emulator is connected through the dedicated debugging interface to the user system. The user system can be debugged under the conditions similar to the actual application conditions. The emulator enables debugging anywhere indoors or out. The host computer for controlling the emulator must be an IBM PC compatible machine with USB.

This manual describes the contents that comply with versions of the High-performance Embedded Workshop 4.0 or later.

Figure 1.1 shows the system configuration using the emulator.

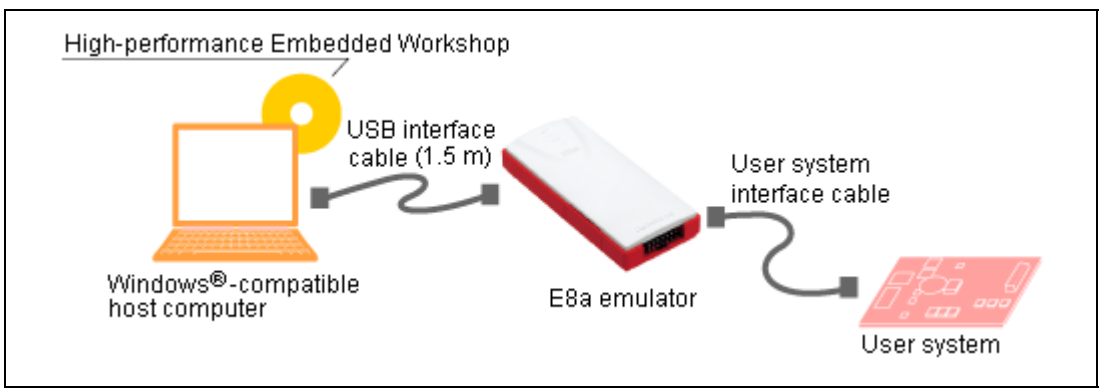


Figure 1.1 System Configuration with the Emulator

The emulator has three activation modes:

(1) Erase Flash and Connect

This mode is used to erase data of the MCU flash memory and activate the debugger. A program for the emulator is also written to at the same time.

(2) Keep Flash and Connect

This mode is used to activate the debugger while retaining data of the MCU flash memory. However, areas for the emulator program and vector areas used by the emulator will be changed.

(3) Program Flash

This mode is used to activate the emulator as a simple programmer. Only the user program is written to when the flash memory is downloaded (the emulator program is not written to). Accordingly, in this mode, program debugging is not available.

The emulator provides the following features:

(1) Support for programming of Renesas' 8- to 32-bit microcomputers

The emulator supports Renesas' 8- to 32-bit microcomputers as the programming tool for the flash memory.

(2) Support for low voltage

The lower limit of the operating voltage is reduced to 1.8 V. The emulator also supports the microcomputer even if its programming voltage for the flash memory is low (reprogramming of the flash memory depends on the voltage to reprogram the flash memory in the microcomputer).

(3) Excellent cost-performance card emulator

Compactness and USB connection are obtained.

(4) Realtime emulation

Realtime emulation of the user system is enabled at the maximum operating frequency of the MCU.

(5) Excellent operability

Using the High-performance Embedded Workshop on the Microsoft® Windows® 2000 or Microsoft® Windows® XP operating system enables user program debugging using a pointing device such as a mouse.

(6) Various debugging functions

Various break and trace functions enable efficient debugging. Breakpoints and break conditions can be set by the specific window, trace information can be displayed on a window, and command-line functions can be used.

(7) Debugging of the user system in the final development stage

The user system can be debugged under conditions similar to the actual application conditions.

(8) Compact debugging environment

A laptop computer can be used as a host computer, creating a debugging environment in any place.

1.1 Warnings

CAUTION

READ the following warnings before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

(1) Check all components against the component list provided at the rear side of the component box after unpacking the emulator.

(2) Never place heavy objects on the casing.

(3) Protect the emulator from excessive impacts and stresses. For details, refer to section 1.2, Environmental Conditions.

(4) When moving the host computer or user system, take care not to vibrate or damage it.

(5) After connecting the cable, check that it is connected correctly. For details, refer to section 3, Preparation before Use.

(6) Supply power to the connected equipment after connecting all cables. Cables must not be connected or removed while the power is on.

1.2 Environmental Conditions

CAUTION

Observe the conditions listed in tables 1.1 and 1.2 when using the emulator. Failure to do so will cause illegal operation in the user system, the emulator product, and the user program.

Table 1.1 Environmental Conditions

Item	Specifications
Temperature	Operating: +10°C to +35°C Storage: –10°C to +50°C
Humidity	Operating: 35% RH to 80% RH, no condensation Storage: 35% RH to 80% RH, no condensation
Vibration	Operating: 2.45 m/s^2 max. Storage: 4.9 m/s^2 max. Transportation: 14.7 m/s^2 max.
Ambient gases	No corrosive gases may be present

Table 1.2 Operating Environments

Item	Description
Host computer	IBM PC or compatible machine with built-in Pentium® III or higher-performance CPU (600 MHz or higher recommended)
OS	Windows® 2000 or Windows® XP
Minimum memory capacity	128 Mbytes or more (double of the load module size)
Hard-disk capacity	Installation disk capacity: 100 Mbytes or more. (Prepare an area at least double the memory capacity (four-times or more recommended) as the swap area.)
Interface	USB (USB1.1, full-speed)*
Pointing device such as mouse	Connectable to the host computer; compatible with Windows® 2000 or Windows® XP.
Power voltage	5.0 ± 0.25 V (USB-bus power type)
Current consumption	500 mA (max)
CD drive	Required to install the software for the emulator or refer to the emulator user's manual.

Note: Can be connected to a host machine supporting USB 2.0.
Not all hardware (such as host machine, USB devices, and USB hub) combination will work and guaranteed.

1.3 Components

Check that all of the components are present when unpacking the product. For details on the E8a emulator components, refer to the component list provided at the rear side of the component box. If all of the components are not present, contact your nearest Renesas sales office.

Section 2 E8a Emulator Functions

This section describes the emulator functions. They differ according to the device supported by the emulator.

2.1 Overview

Table 2.1 gives a functional overview of the emulator.

Table 2.1 Emulator Functions

No.	Item	Function
1	User program execution function	• Executes a program with the operating frequency within a range guaranteed by devices. • Reset emulation • Step functions: Single step (one step: one instruction) Source-level step (one step: one source) Step over (a break did not occur in a subroutine) Step out (when the PC points to a location within a subroutine, execution continues until it returns to the calling function)
2	Reset function	• Issues a reset from the High-performance Embedded Workshop to the device during break.
3	Trace function	• Branch trace function incorporated in the device (four branches) The support of this function depends on a device.
4	Break functions	• Hardware break condition (one condition) The support of this function depends on a device. • PC break condition (255 points) • Forced break function • Address match break function

Table 2.1 Emulator Functions (cont)

No.	Item	Function
5	Memory access function	• Downloading to RAM • Downloading to flash memory • Single-line assembly • Reverse assembly (disassembly) • Reading of memory • Writing to memory • Automatic updating of a display of selected variables during user program execution • FILL • Search • Move • Copy
6	General/control register access function	Reads or writes the general/control register.
7	Internal I/O register access function	Reads or writes the internal I/O register.
8	Source-level debugging function	Various source-level debugging functions.
9	Command line function	Supports command input. Batch processing is enabled when a file is created by arranging commands in input order.
10	Start/Stop function	Executes the specified routine of the user program immediately before starting and immediately after halting program execution.
11	Help function	Describes the usage of each function or command syntax input from the command line window.

The specific functions of the emulator are described in the next section.

2.2 Trace Functions

The branch source addresses, mnemonics, operands, and source lines are displayed. Since this function uses the trace buffer built into the device, a realtime trace can be acquired.

Note: The MCUs which have no trace function cannot use this function.

2.3 Break Functions

The E8a emulator has the following four break functions.

(1) Hardware break function

H8 R8C series

Uses a break controller incorporated in the device.

The access address, instruction fetch address, data, or bus cycle condition can be set.

This function can also be set from the [Event] column in the [Source] window. For the setting, refer to section 5.3.1 Address Interrupt Breakpoints.

Note: The MCUs which have no hardware break function cannot use this function.

(2) PC break function (BREAKPOINT)

Breaks when the dedicated instruction at the specified address that has been replaced is executed.

This function can be set on the [Breakpoint] page in the [Eventpoint] dialog box. It can also be set when the [S/W Breakpoints] column for the line to be set is double-clicked in the [Source] or [Disassembly] window.

(3) Forced break function

Forcibly breaks the user program.

(4) Address match break function

Stops the target program immediately before a specified address instruction is executed. This function is featured by the address match interrupt of the MCU. This function can be set from the [Event] column in the Source window. For the setting, refer to section 5.2.2, Viewing the Source Code.

Note: The address match break points vary depending on the device. And, user program operation when device's address match interrupts are used is not guaranteed.

2.4 Memory Access Functions

The emulator has the following memory access functions.

(1) Memory read/write function

[Memory] window: The memory contents are displayed in the window. Only the amount specified when the [Memory] window is opened can be read. If the memory is written in the [Memory] window, a read in the range displayed in the [Memory] window will occur for updating the window. When the [Memory] window is not to be updated, change the setting in [Lock Refresh] from the popup menu.

me command: A command line function that reads or writes the specified amount of memory at the specified address.

(2) User program downloading function

A load module registered in the workspace can be downloaded. Such module can be selected from [Download Modules] in the [Debug] menu. Downloading is also possible by a popup menu that is opened by right-clicking on the mouse at the load module in the workspace. The user program is downloaded to the RAM or flash memory.

This function can download debugging information required for source-level debugging with the Elf/Dwarf2 or IEEE695 file having the debugging information.

(3) Memory data uploading function

The specified amount of memory from the specified address can be saved in an S-format file.

(4) Memory data downloading function

The memory contents saved in the S-type-formatted file can be downloaded. Select [Load] from the popup menu in the [Memory] window.

(5) Displaying the variable contents

The variable contents specified in the user program are displayed. For the usage of the function for displaying the variable contents, refer to High-performance Embedded Workshop User's Manual.

(6) Other memory operation functions

Other functions are as follows:

- Memory fill
- Memory copy
- Memory save
- Memory verify
- Memory search
- Internal I/O display
- Displaying label and variable names and their contents

Notes: 1. Memory access during user program execution:
When memory is accessed from the memory window, etc. during execution of the user program, execution stops for the memory access and is then resumed. Therefore, realtime emulation cannot be performed.

2. Memory access during user program break:
The program can also be downloaded and the BREAKPOINT can be set for the flash memory area by the emulator.

2.5 Stack Trace Function

The emulator uses the information on the stack to display the names of functions in the sequence of calls that led to the function to which the program counter is currently pointing. This function can be used only when the load modules that have the following formats are loaded:

- M32C family C complier (M3T-NC308WA): IEEE695
- M16C family C compiler (M3T-NC30WA): IEEE695
- R8C series C compiler (M3T-NC8C): IEEE695
- H8,H8S,H8SX family C/C++ compiler (such as [R0C40008XSW06R]): Elf/Dwarf2

For the usage of this function, refer to section 6.19, Stack Trace Function.

2.6 Start/Stop Function

The emulator executes the specified routine of the user program immediately before starting and immediately after halting program execution. This function is used to control the user system in synchronization with execution and halting of the user program.

For the usage of this function, refer to section 5.6, Using the Start/Stop Function.

Note: The start/stop function is not supported in the H8/Tiny or H8/Super Low Power series, the 740 family, or the R8C/10, R8C/11, R8C/12, or R8C/13 group.

2.7 Online Help

An online help explains the usage of each function or the command syntax that can be entered from the command line window.

Select [Emulator Help] from the [Help] menu to view the emulator help.

Section 3 Preparation before Use

3.1 Emulator Preparation

Unpack the emulator and prepare it for use as follows:

⚠ WARNING

READ the reference sections shaded in Figure 3.1 before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

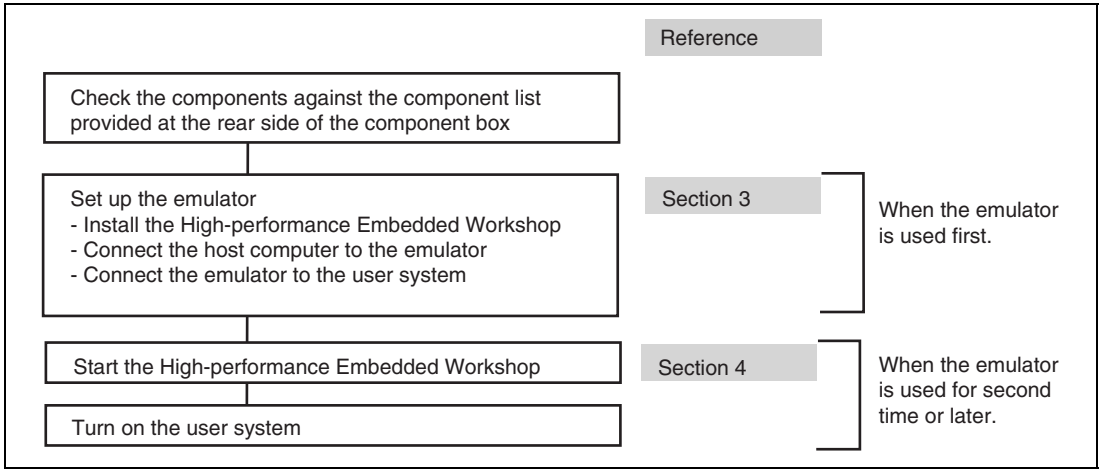

Reference
Check the components against the component list provided at the rear side of the component box
Set up the emulator
- Install the High-performance Embedded Workshop
- Connect the host computer to the emulator
- Connect the emulator to the user system
Section 3
When the emulator is used first.
Start the High-performance Embedded Workshop
Turn on the user system
Section 4
When the emulator is used for second time or later.

Figure 3.1 Emulator Preparation Flow Chart

3.2 Emulator Hardware Configuration

As shown in Figure 3.2, the emulator consists of an emulator, a USB cable, and a user system interface cable. The emulator is connected to the host computer via USB 1.1, and also to the USB port conforming to USB 2.0.

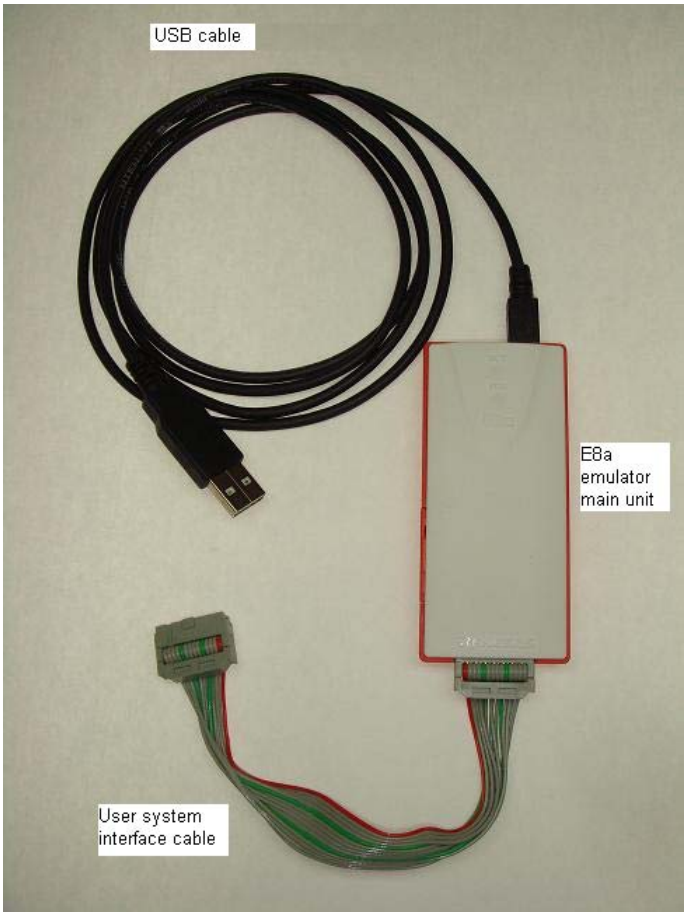


Figure 3.2 Emulator Hardware Configuration

The names of each section of the emulator are explained next.

Emulator Upper-side Panel:

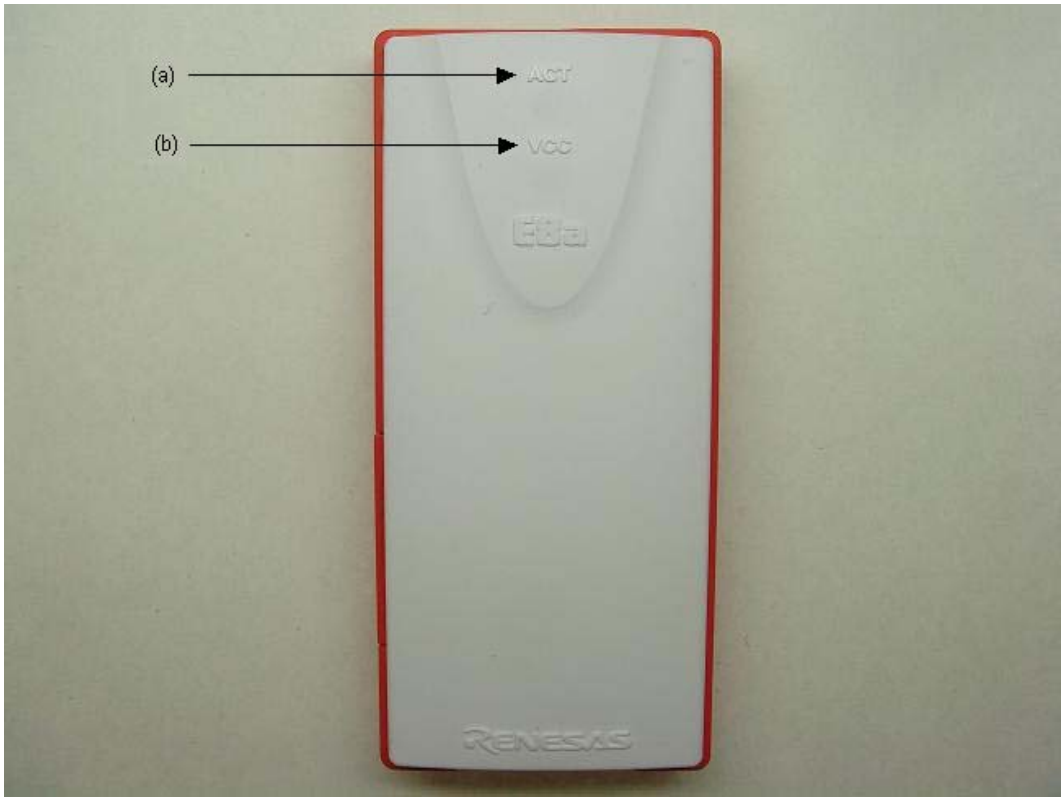


Figure 3.3 Emulator Upper-side Panel

(a) ACT LED: An LED to show the operating state of the emulator software. When this LED is lit (yellow-green), the E8a control software is in operation.

(b) VCC LED: An LED to show the power state of the user system. When power is supplied from the emulator to the user system, this LED is lit as orange. When power is supplied from the user system to the emulator, this LED is lit as yellow-green.

Emulator Host-Computer-Side Panel:

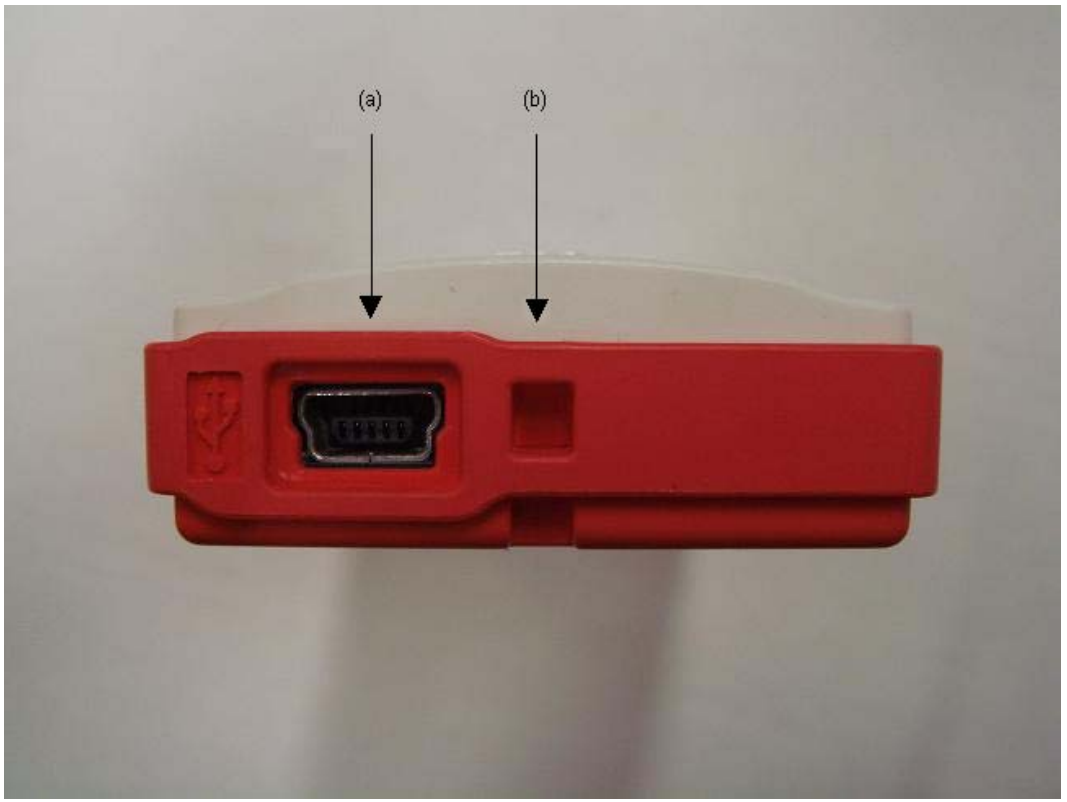


Figure 3.4 Emulator Host-Computer-Side Panel

(a) Host-side connector: A USB cable connector to connect the emulator to the host computer. Be sure to connect the provided USB cable.

(b) Portable strap hole: A hole for the strap (No strap is provided for this product).

Emulator User-System-Side Panel:

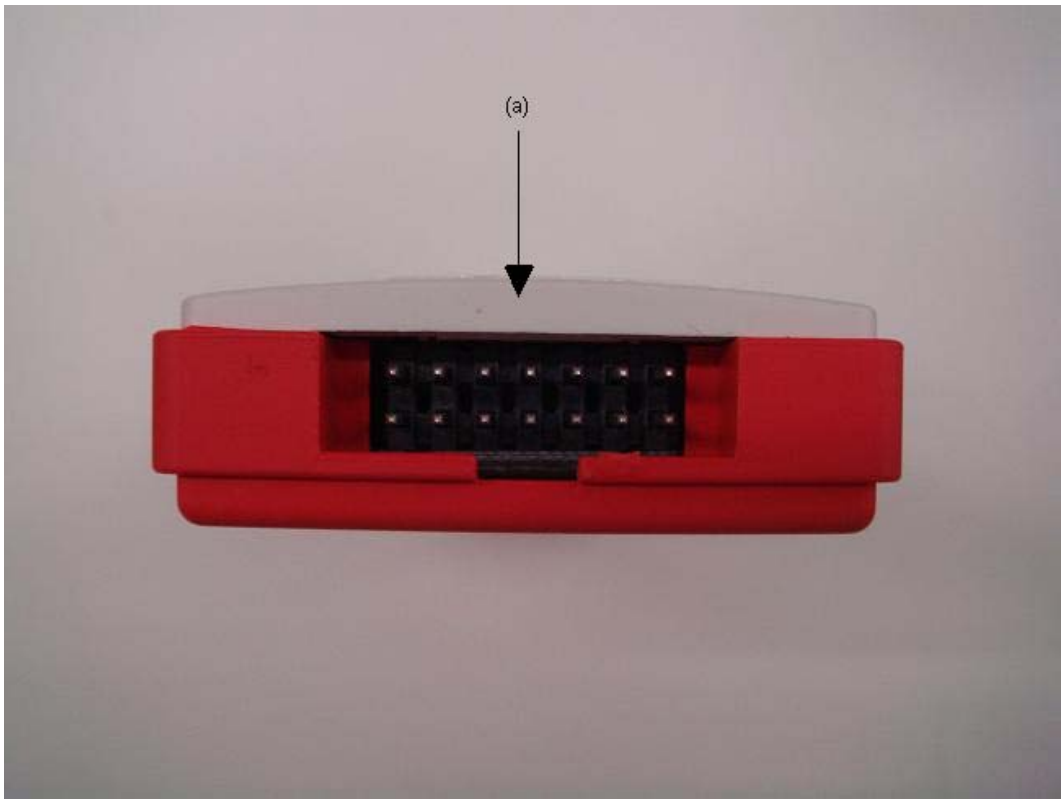


Figure 3.5 Emulator User-System-Side Panel

(a) User-system-side connector: A user system interface cable is connected.

Emulator Bottom Panel:

Figure 3.6 Emulator Bottom Panel

(a) Seal for product management: The serial number and a description of the emulator are written on the seal. The actual written content will depend on when you purchased the product.

Emulator Left-Side Panel:

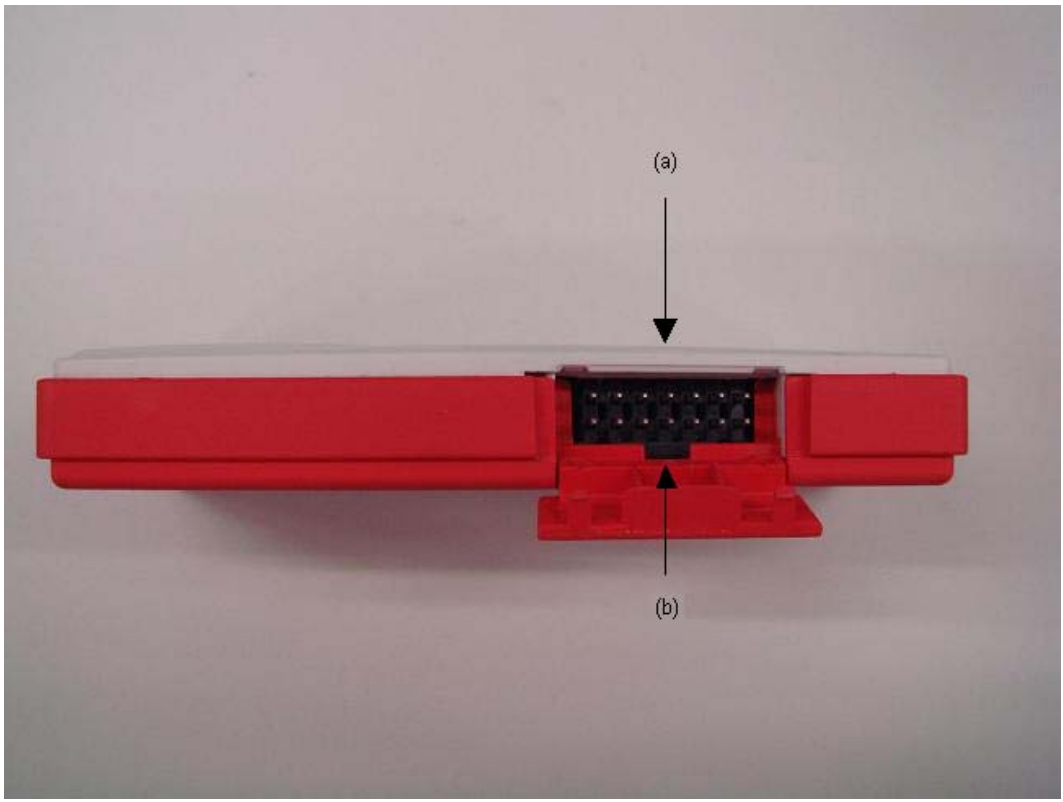


Figure 3.7 Emulator Left-Side Panel

(a) Self-check connector: A connector for the user system interface cable when the E8aSCP (self-check program) is run.

(b) Cover for self-check connector: A cover for the self-check connector. Close this cover when the E8aSCP is not run.

3.3 Emulator Software Configuration

The following item is included with the emulator debugger when it is purchased:

- E8a Emulator Software: R0E00008AKCE00SR (CD)

3.3.1 CD

Install the software as instructed by the displayed messages by inserting the included CD into the CD-ROM drive. The folders contain the files and programs listed below.

Table 3.1 Contents of the CD Directories

Directory Name	Contents	Description
Dlls	Microsoft® runtime library	A runtime library for the High-performance Embedded Workshop. The version is checked at installation and this library is copied to the hard disk as part of the installation process.
Drivers	E8a emulator driver	The E8a emulator drivers.
Help	Online help for the E8a emulator	An online help file
Manuals	E8a emulator manual	E8a emulator user's manual. This is provided as a PDF file.

3.4 Installing Emulator’s Debugger

When the CD is inserted in the host computer’s CD drive, execute HewInstMan.exe from the root directory of the CD.

3.5 Connecting the Emulator to the Host Computer

This section describes how to connect the emulator to the host computer. For the position of each connector of the emulator, refer to section 3.2, Emulator Hardware Configuration.

Notes: 1. When [Add New Hardware Wizard] is displayed, select the [Search for the best driver for your device. (Recommended)] radio button and then the [Specify a location] check box to select the path to be searched for drivers. The location must be specified as <Drive>:\DRIVERS. (<Drive> is the CD drive name.)

2. Be sure to install the emulator's debugger before putting the emulator in place.

⚠ WARNING

Always switch OFF the emulator product and the user system before connecting or disconnecting any CABLES except for the USB interface cable. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY.
The USER PROGRAM will be LOST.

The emulator is connected to the host computer via the USB 1.1, and also to the USB port conforming to USB 2.0. Figure 3.8 shows the system configuration.

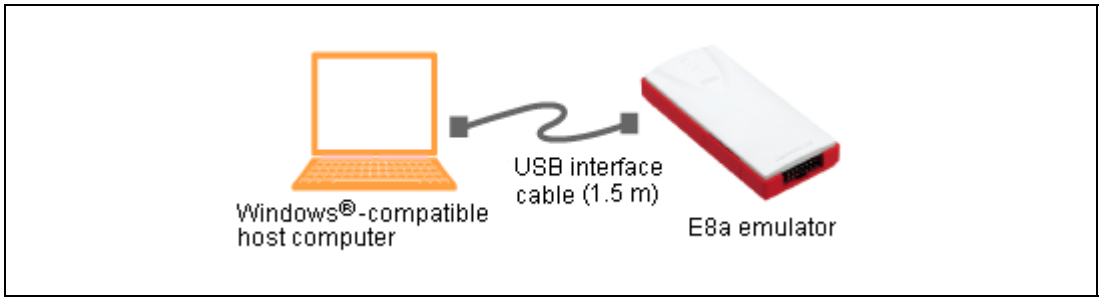


Figure 3.8 System Configuration when Connecting the Emulator to the Host Computer

3.6 Connecting the Emulator to the User System

Use the procedure below to connect the emulator to the user system with the user system interface cable, or to disconnect them when moving the emulator or the user system.

1. Check that the host computer is turned off or the emulator is not connected to the host computer with the USB cable.
2. Connect the user system interface cable to the user-side connector of the emulator.
3. Connect the USB cable to the host-computer-side connector of the emulator.

(1) The connector must be installed on the user system. Table 3.2 shows the recommended connector for the emulator.

Table 3.2 Recommended Connector

Type Number	Manufacturer	Specifications
2514-6002	3M Limited	14-pin straight type

Note: When the connector is installed on the user system, do not install any components within 3 mm of the connector.

(2) The pin assignments of the connector are shown in the section of ‘Pin Assignments of the E8/E8a Connector’ in the additional documents for the E8a emulator user’s manual.

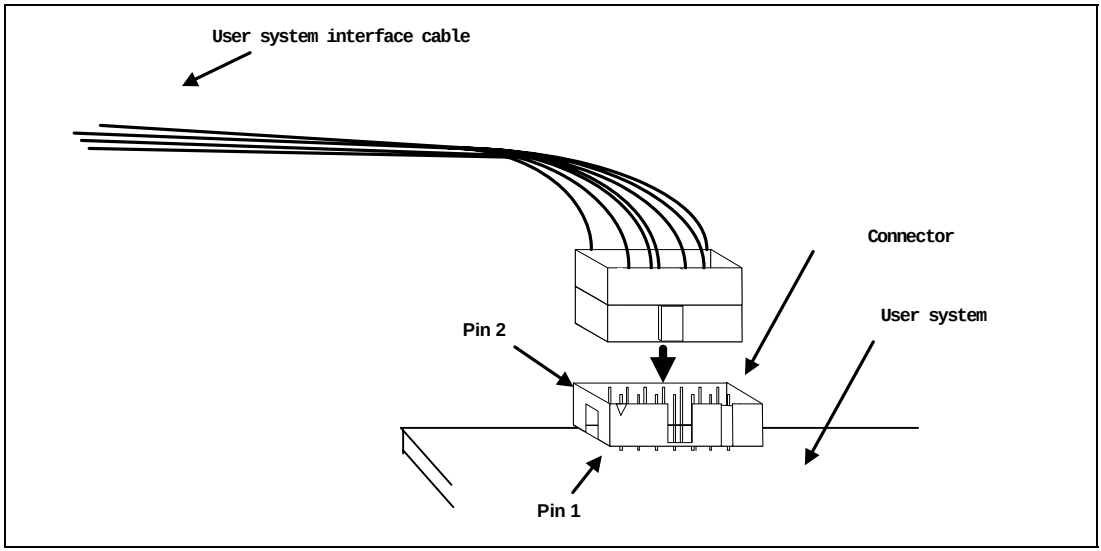


Figure 3.9 Connecting the User System Interface Cable to the User System

Notes:
1. To connect the signals output from the E8a connector, refer to the MCU pin alignment.
2. To remove the user system interface cable from the user system, pull the connector upward.
3. The range of frequencies that the emulator operates at is different according to the MCUs used.
4. Connect the signals from the E8a connector as shown in section 3 or section 4 in the additional documents for the E8a emulator user’s manual.

3.7 Connecting System Ground

⚠ WARNING

Separate the frame ground from the signal ground at the user system. Failure to do so will result in malfunction of the emulator.

The emulator's signal ground is connected to the user system's signal ground. In the emulator, the signal ground and frame ground are connected. In the user system, connect the frame ground only; do not connect the signal ground to the frame ground (Figure 3.10).

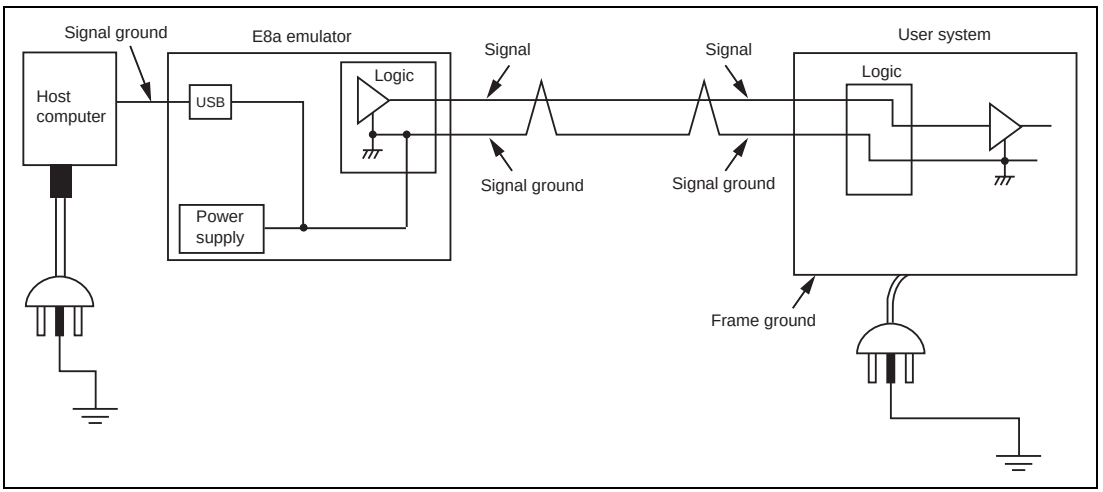


Figure 3.10 Connecting System Ground

3.8 System Check

When the software is executed, use the procedure below to check that the emulator is connected correctly. Here, use the workspace for a tutorial provided on the product.

Refer to section 4, Preparations for Debugging, for the other activating method to create a new project or use a workspace for the High-performance Embedded Workshop of the old version.

1. Connect the emulator to the host computer.
2. Connect the user system interface cable to the connector of the emulator.
3. Connect the user system interface cable to the connector in the user system.
4. Select [Renesas] -> [High-performance Embedded Workshop] -> [High-performance Embedded Workshop] from [Programs] in the [Start] menu of Windows®
5. The [Welcome!] dialog box is displayed.

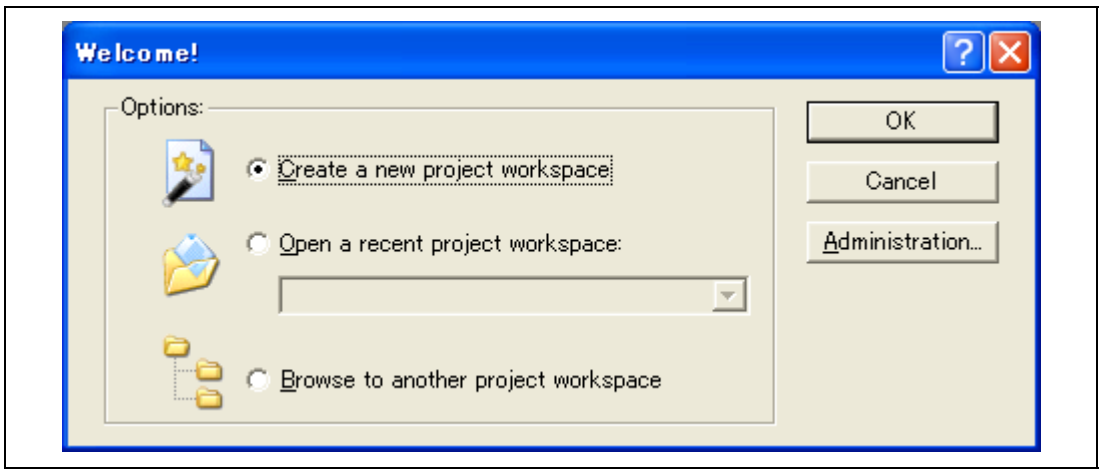


Figure 3.11 [Welcome!] dialog box

[Create a new project workspace] radio button:	Creates a new workspace.
[Open a recent project workspace] radio button:	Uses the current workspace and displays the history of the opened workspace.
[Browse to another project workspace] radio button:	Uses the current workspace; this radio button is used when the history of the opened workspace does not remain

To use a workspace for the tutorial, select the [Browse to another project workspace] radio button and click the [OK] button.

When the [Open workspace] dialog box is opened, specify the following directory:
<OS installation directory>

\WorkSpace\Tutorial\E8\xxxx\Tutorial
After the directory has been specified, select the following file and click the [Open] button.

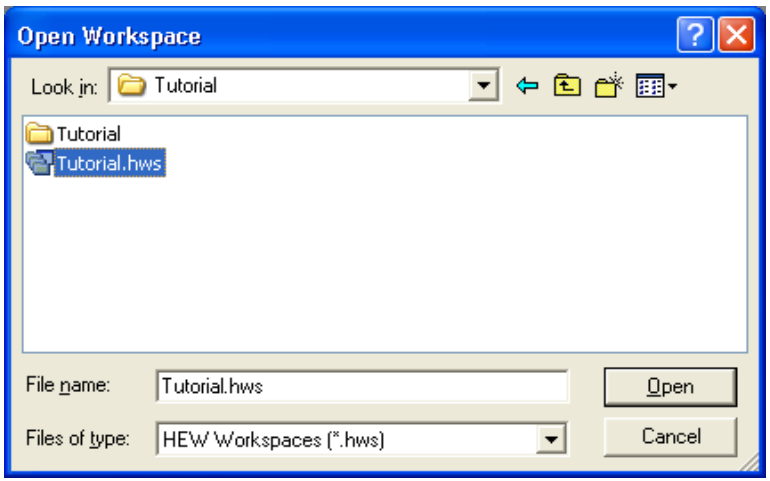


Figure 3.12 [Open Workspace] Dialog Box

Note:
The directory for a tutorial differs depending on the device.

- M32C
  Drive where the OS has been installed:
  \WorkSpace\Tutorial\E8a\M32C\Tutorial
- M16C
  Drive where the OS has been installed:
  \WorkSpace\Tutorial\E8a\M16C\Tutorial
- R8C/Tiny
  Drive where the OS has been installed:
  \WorkSpace\Tutorial\E8a\R8C\Tutorial
- 740
  Drive where the OS has been installed:
  \WorkSpace\Tutorial\E8a\740\Tutorial
- H8/Tiny normal mode
  Drive where the OS has been installed:
  \WorkSpace\Tutorial\E8a\H8\Tutorial
- H8/Tiny advanced mode
  Drive where the OS has been installed:
  \WorkSpace\Tutorial\E8a\H8\TutorialADV

- H8/Super Low Power
  Drive where the OS has been installed:
  \WorkSpace\Tutorial\E8a\H8\TutorialSLP

6. The [Emulator Setting] dialog box is displayed.

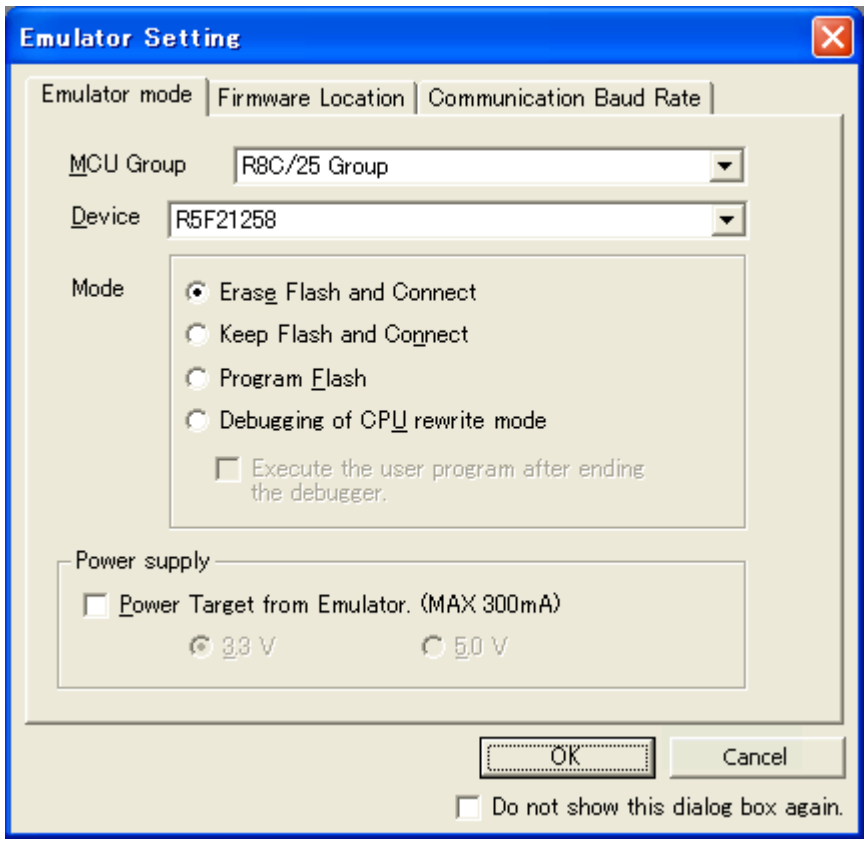


Figure 3.13 [Emulator Setting] Dialog Box

Select the device name in use from the [MCU Group] drop-down list box.

Note: The [MCU Group] drop-down list box is not displayed in the H8/Tiny or H8/Super Low Power series.

Select the device name in use from the [Device] drop-down list box.

The following items are selected in the [Mode] group box.

— Erase Flash and Connect
  This mode is used to erase data of the MCU flash memory and activate the debugger. A program for the emulator is also written to at the same time.

Note: The emulator occupies the user's flash memory area according to the target MCUs. For details, refer to the additional document, Notes on Connecting the xxxxx.

— Keep Flash and Connect

This mode is used to activate the debugger while retaining data of the MCU flash memory. However, areas for the emulator program and vector areas used by the emulator will be changed.

— Program Flash

This mode is used when the emulator is used as a flash memory programmer. It cannot be used for debugging programs.
Register the load module in the workspace to download it.

Note: It is necessary to input the ID code of the flash memory by the target device. For details, refer to the additional document, Notes on Connecting the xxxxx.

— Debugging of CPU rewrite mode

This mode is used to debug the program that rewrites the CPU. In this mode, the following debug operations, which require reprogramming of the flash memory, are not available:
- Setting the PC breakpoint
- Changing the memory contents in the flash memory area

The debugger is activated by erasing data of the MCU flash memory. A program for the emulator program is also written to at the same time.

Note: This mode is not supported in the H8/Tiny or H8/Super Low Power series.

When the [Execute the user program after ending the debugger.] check box is selected, the user program is executed at the same time as the debugger operation is ended with the emulator connected to the user system. This check box is only available when the [Program Flash] mode has been selected.

When the [Power Target from Emulator. (MAX 300mA)] check box is selected, power will be supplied to the user system up to 300 mA. Then select 3.3 V or 5.0 V according to the power voltage of the user system.

CAUTION

Before carrying out the power supply, check the power specification of the user system and there is no short circuit in the user system. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

7. The [Connecting] dialog box is displayed and the emulator connection is started.

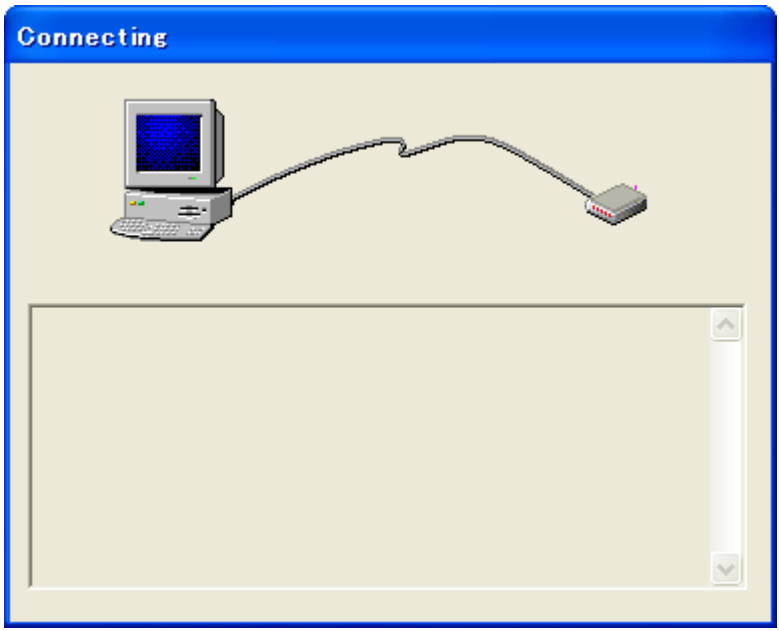


Figure 3.14 [Connecting] Dialog Box

8. When the version of the E8a firmware downloaded into the emulator is old, the message box shown in Figure 3.15 will be displayed.

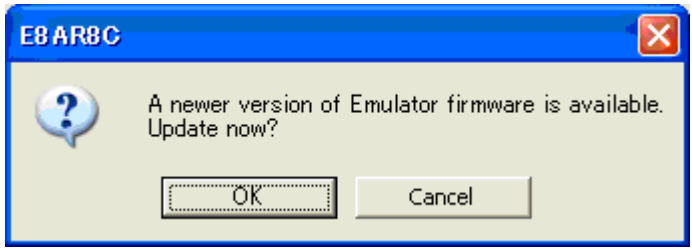


Figure 3.15 Dialog Box for confirming the E8a firmware download

Note: When [OK] button is clicked, it starts downloading the E8a firmware. Do not connect/disconnect the USB cable until the download status dialog box closes.

9. When [Power Target from Emulator. (MAX 300mA)] is not checked, the dialog box shown in Figure 3.16 will be displayed.

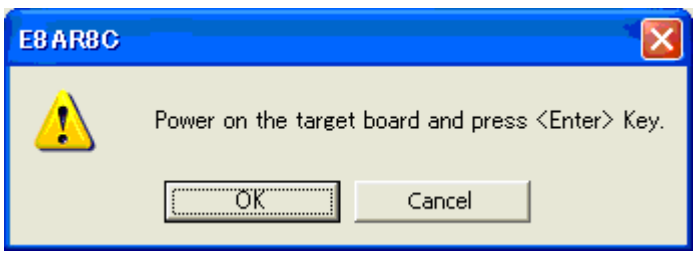


Figure 3.16 Dialog Box of the Power-on Request Message

10. If no power is supplied to the user system, the dialog box for confirming the power supply status of the user system will be displayed.

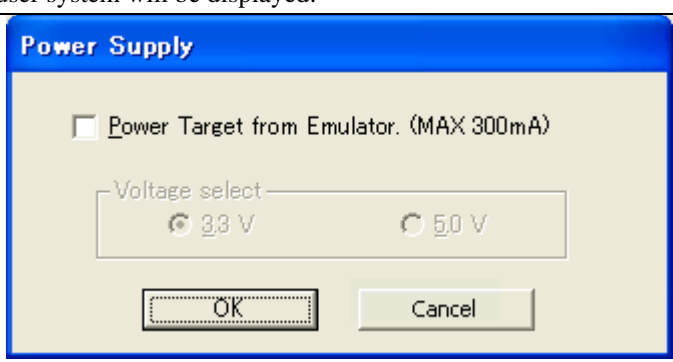


Figure 3.17 [Power Supply] Dialog Box

When no power is supplied from the emulator, click the [OK] button without selecting the [Power Target from Emulator. (MAX 300mA)] check box.

When power is supplied from the emulator, select the [Power Target from Emulator. (MAX 300mA)] check box and select 3.3 V or 5.0 V according to the power voltage of the user system.

CAUTION

Before carrying out the power supply, check the power specification of the user system and there is no short circuit in the user system. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

11. When "Connected" is displayed in the [Output] window of the High-performance Embedded Workshop, the emulator initiation is completed

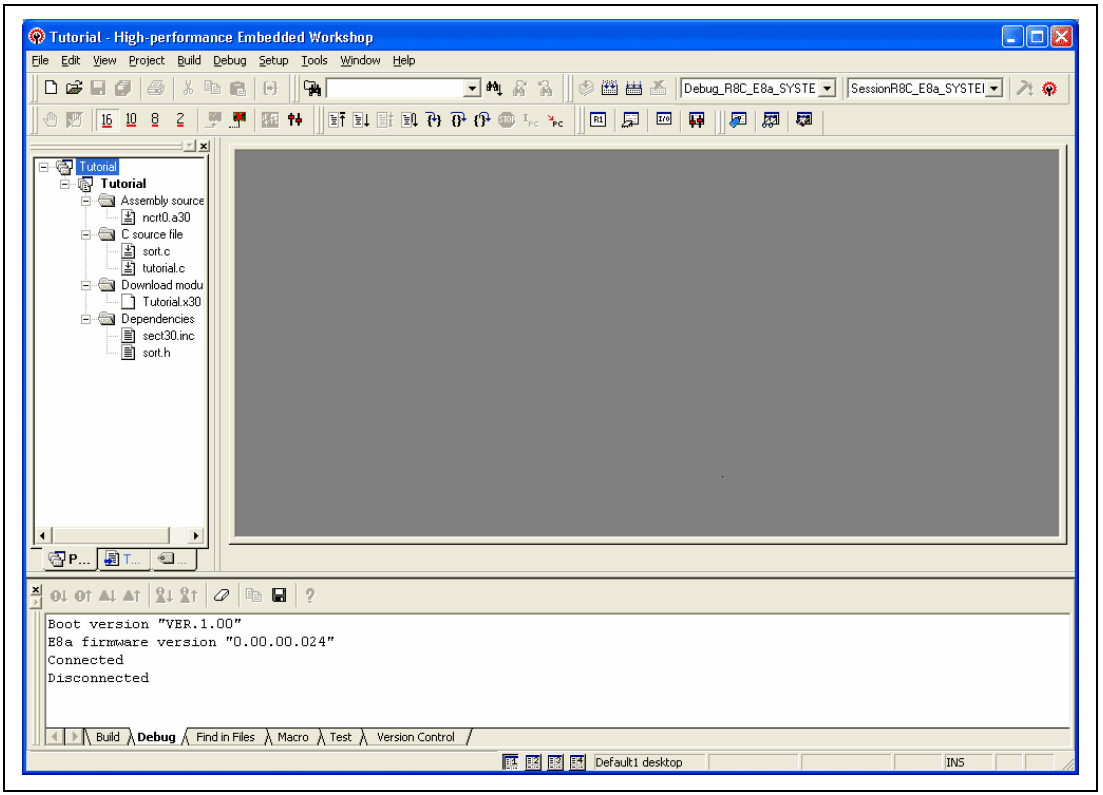


Figure 3.18 High-performance Embedded Workshop Window

Note: When the user program has already been downloaded to the flash memory, source-level debugging cannot be executed because there is no debugging information on the user program after the emulator has been activated. Be sure to load the debugging information file. For details, refer to section 4.2.1, Setting at Emulator Activation.

Notes: 1. If the user system interface cable is disconnected from the connector on the user system, the following dialog box will appear.

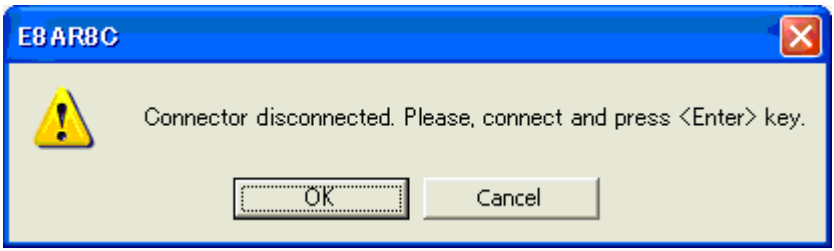


Figure 3.19 [Connector disconnected] Dialog Box

2. If the emulator is not properly initialized, one of the dialog boxes shown in figures 3.20 through 3.24 will appear.

(a) The following dialog box is displayed when the flash memory cannot be erased. Exchange the MCU since the flash memory has been programmed more times than the limitation.

Figure 3.20 [Flash memory erase error!] Dialog Box

(b) The following dialog box is displayed when the flash memory cannot be programmed. An incorrect system clock value has been input or the flash memory has been programmed more times than the limitation.

Figure 3.21 [Error sending Flash memory write program] Dialog Box

(c) The following dialog box is displayed when an incorrect ID code has been input. For the H8/Tiny, H8/Super Low Power, R8C/10, 11, 12 and 13, if an ID code does not match, the flash memory will be completely erased.

Figure 3.22 [ID code error!] Dialog Box

(d) The following dialog box is displayed when the MCU cannot communicate with the emulator. Check the MCU settings.

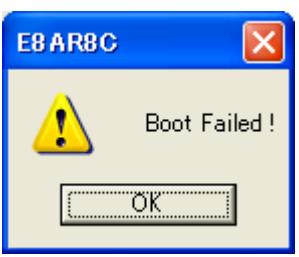


Figure 3.23 [Boot Failed!] Dialog Box

3. If an incorrect driver has been selected, the following dialog box will appear.

Figure 3.24 [Unable to restore the previous driver settings] Dialog Box

3.9 Uninstalling the Emulator's Debugger

Follow this procedure to remove the installed emulator's debugger from the user's host computer. As the installed product is known by the High-performance Embedded Workshop, uninstall the product on the High-performance Embedded Workshop screen.

1. Activate the High-performance Embedded Workshop.
2. Click the [Administration…] button in the [Welcome!] dialog box.

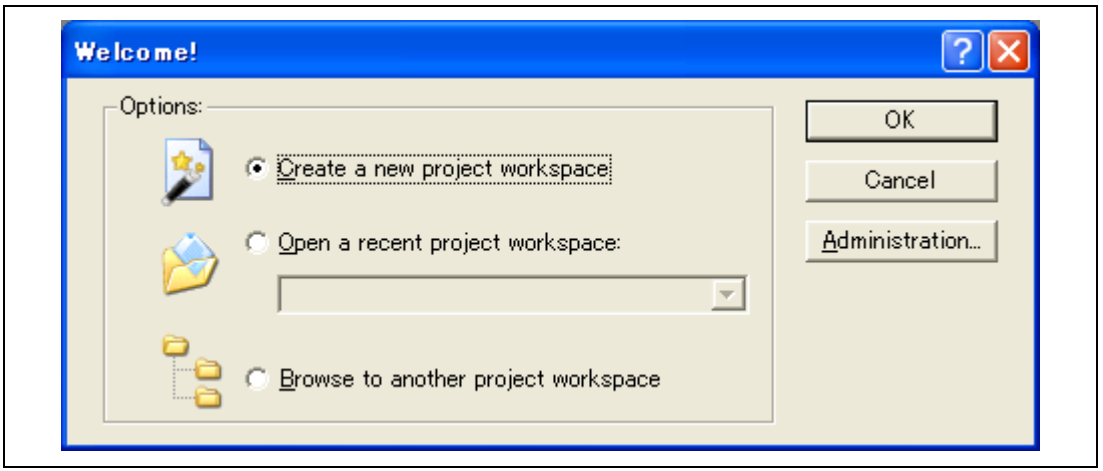


Figure 3.25 [Welcome!] Dialog Box

3. The [Tools Administration] dialog box is opened.

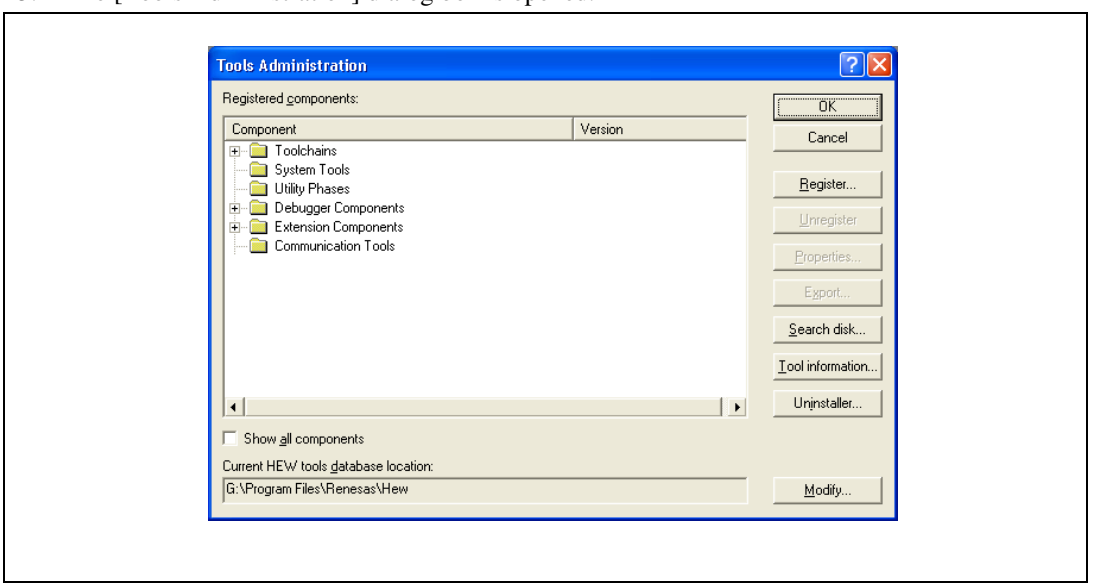


Figure 3.26 [Tools Administration] Dialog Box

Click the [+] mark at the left of [Debugger Components] in the [Registered components] list box to list the installed components. Then, highlight the product name to be uninstalled.

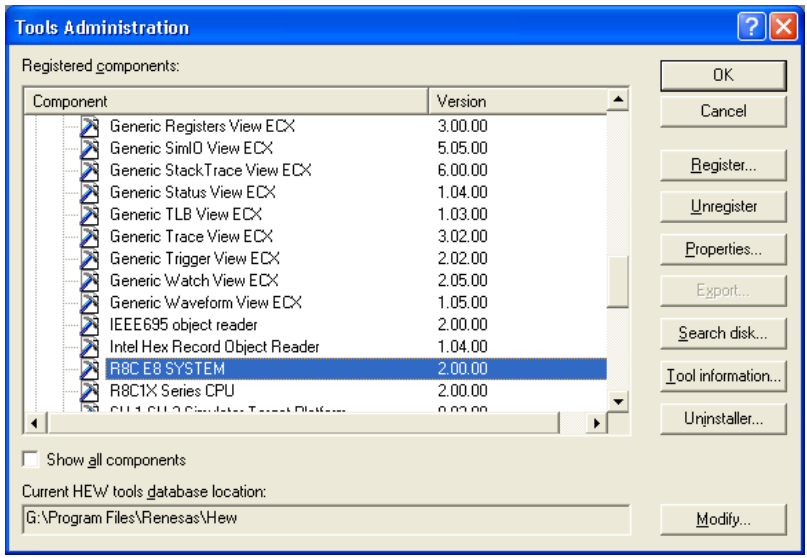


Figure 3.27 Highlighting the Product to be Uninstalled

Click the [Unregister] button. After the following message box is displayed, click the [Yes] button.

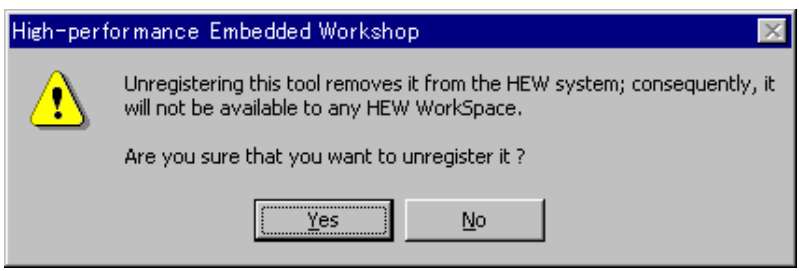

High-performance Embedded Workshop
Unregistering this tool removes it from the HEW system; consequently, it will not be available to any HEW WorkSpace.
Are you sure that you want to unregister it ?
Yes
No

Figure 3.28 [Unregistering this tool] Message Box

This is the end of canceling the High-performance Embedded Workshop registration. Then, remove the file for the emulator from the host computer.

Click the [Uninstaller…] button in the [Tools Administration] dialog box to open the [Uninstall HEW Tool] dialog box.

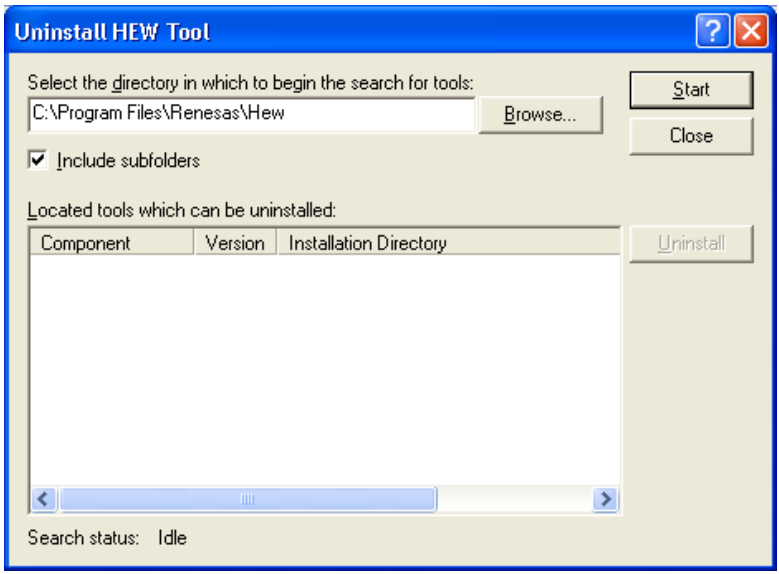


Figure 3.29 [Uninstall HEW Tool] Dialog Box

Click the [Start] button to list the installed components.

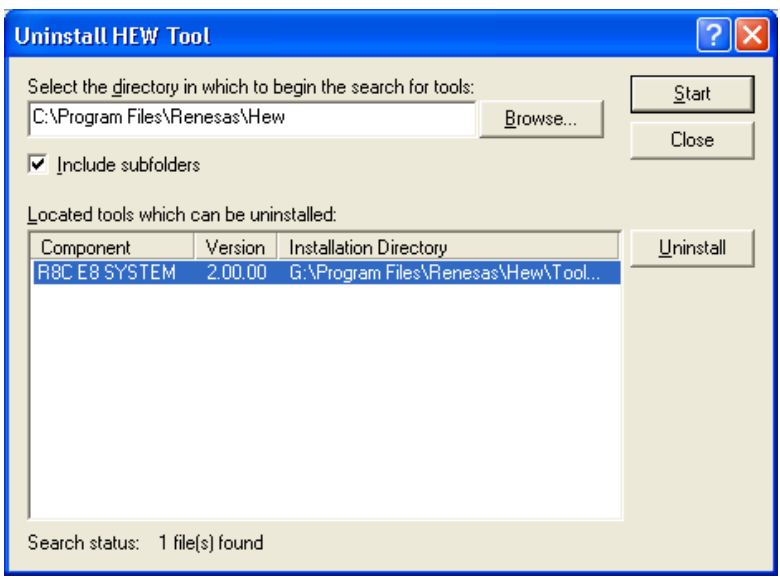


Figure 3.30 Highlighting the Product to be Uninstalled

Highlight the product name to be uninstalled and click the [Uninstall] button. This is the end of uninstallation.

CAUTION

A shared file may be detected while the program is being removed. If another product may be using the shared file, do not remove the file. If another product does not start up after the removal process, re-install that product.

4.1 Method for Activating High-performance Embedded Workshop

To activate the High-performance Embedded Workshop, follow the procedure listed below.

1. Connect the emulator to the host computer and the user system, then turn on the user system.
2. Select [Renesas] -> [High-performance Embedded Workshop] -> [High-performance Embedded Workshop] from [Programs] in the [Start] menu of Windows®
3. The [Welcome!] dialog box is displayed.

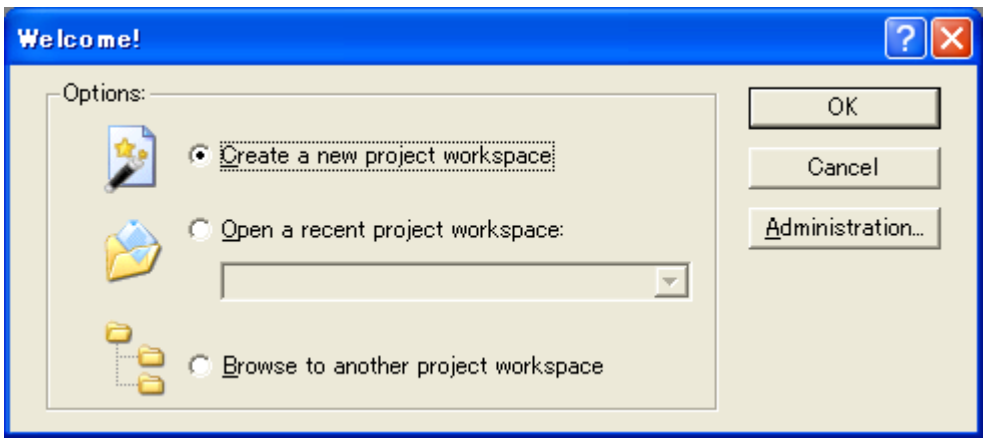


Figure 4.1 [Welcome!] Dialog Box

[Create a new project workspace] radio button:	Creates a new workspace.
[Open a recent project workspace] radio button:	Uses the current workspace and displays the history of the opened workspace.
[Browse to another project workspace] radio button:	Uses the current workspace; this radio button is used when the history of the opened workspace does not remain.

In this section, we describe the following three ways to start up the High-performance Embedded Workshop:

- [Create a new project workspace] - a toolchain is not in use
- [Create a new project workspace] - a toolchain is in use
- [Browse to another project workspace]

The [Open a recent project workspace] radio button is used to omit the operation for specifying the workspace file when [Browse to another project workspace] is selected.

4.1.1 Creating the New Workspace (Toolchain Not Used)

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select [Create a new project workspace] radio button and click the [OK] button.

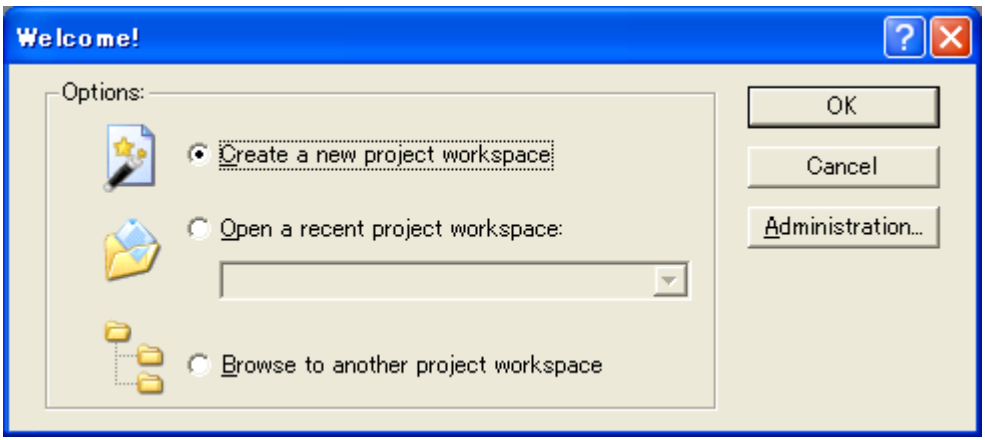


Figure 4.2 [Welcome!] Dialog Box

2. The Project Generator is started. This section omits the description on the setting for the toolchain.
   If you have not purchased the toolchain, the following dialog box is displayed.

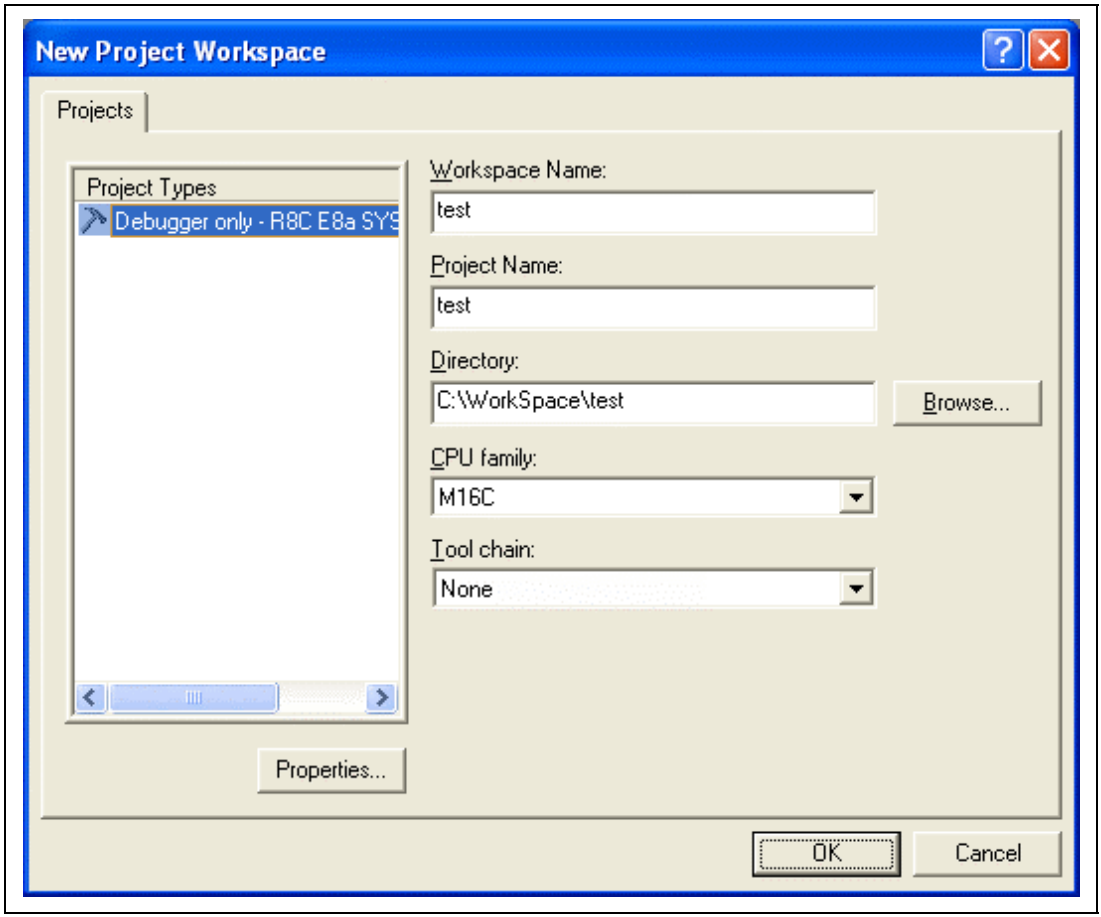


Figure 4.3 [New Project Workspace] Dialog Box

[Workspace Name] edit box: Enter the new workspace name. Here, enter 'test'.

[Project Name] edit box: Enter the project name. When the project name is the same as the workspace name, it needs not be entered.

[CPU family] drop-down list box: Select the target CPU family.
M32C: Select [M16C/80, M32C]
R8C, M16C: Select [M16C].
740: Select [740 Family].
H8 Tiny/Super Low Power: Select [H8S, H8/300].

[Tool chain] drop-down list box: Here, select [None].

Other list boxes are used for setting the toolchain; the fixed information is displayed when the toolchain has not been installed.

3. The following dialog box is displayed.

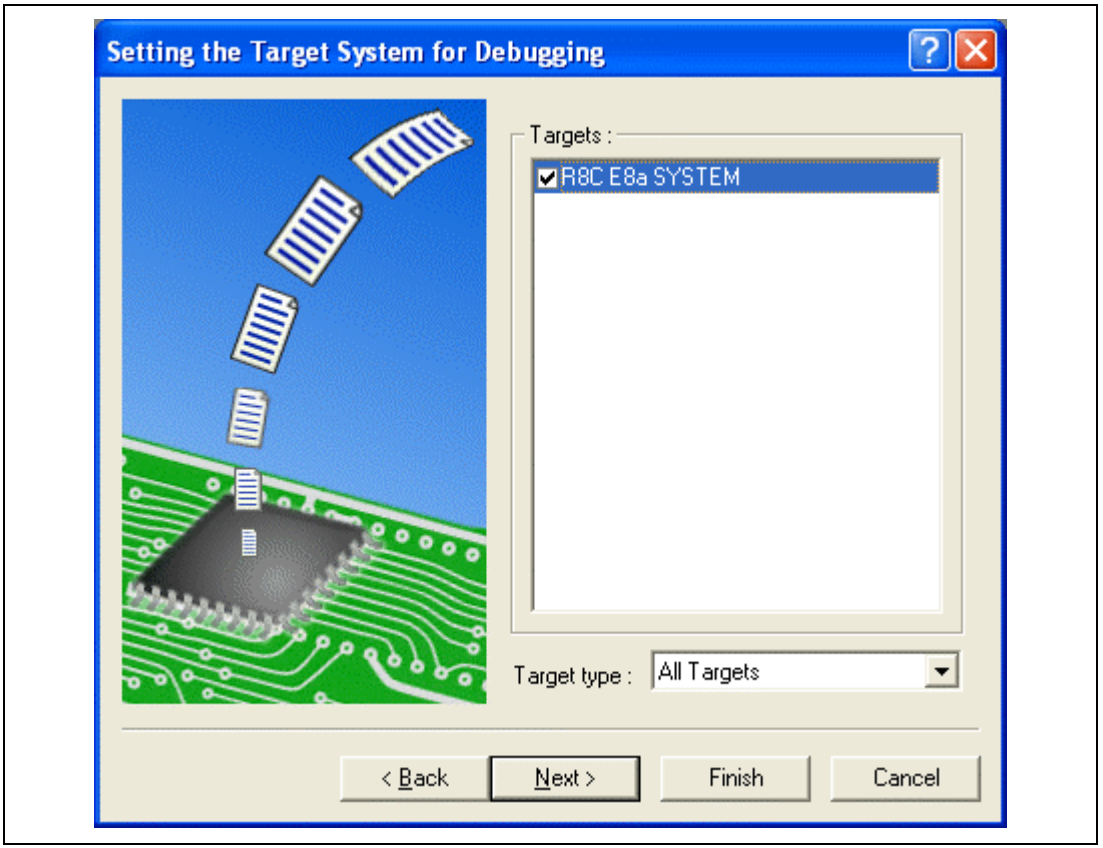


Figure 4.4 [Setting the Target System for Debugging] Dialog Box

Select the target system for debugging and click the [Next] button.

[Target system for debugging]	[Target MCU]
M32C E8a SYSTEM:	M32C
M16C E8a SYSTEM:	M16C
R8C E8a SYSTEM:	R8C/Tiny
740 E8a SYSTEM:	740
H8 Tiny/Super Low Power E8a SYSTEM 300H:	H8/300H Tiny, H8/300H Super Low Power
H8 Tiny/Super Low Power E8a SYSTEM 300L:	H8/300 Super Low Power

4. Set the configuration file name. The configuration file saves the setting of the High-performance Embedded Workshop except the emulator.

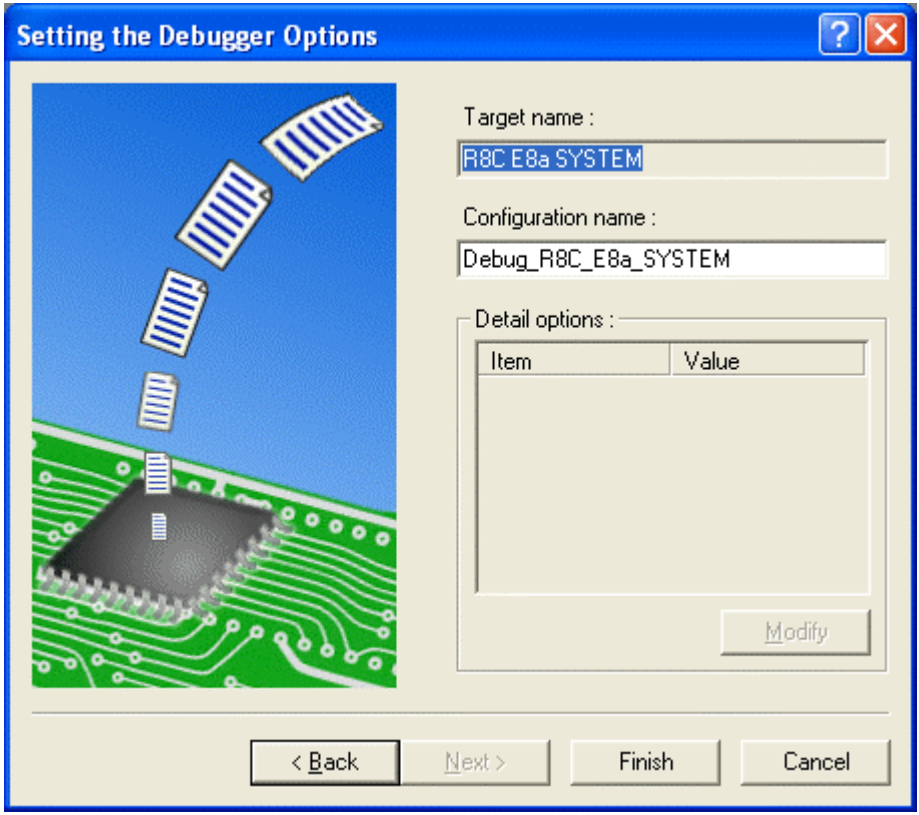


Figure 4.5 [Setting the Debugger Options] Dialog Box

This is the end of the emulator setting.

Click the [Finish] button to exit the Project Generator. The High-performance Embedded Workshop is activated.

5. After the High-performance Embedded Workshop has been activated, the emulator is automatically connected. For operation during connection, refer to section 3.8, System Check.

4.1.2 Creating the New Workspace (Toolchain Used)

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select [Create a new project workspace] radio button and click the [OK] button.

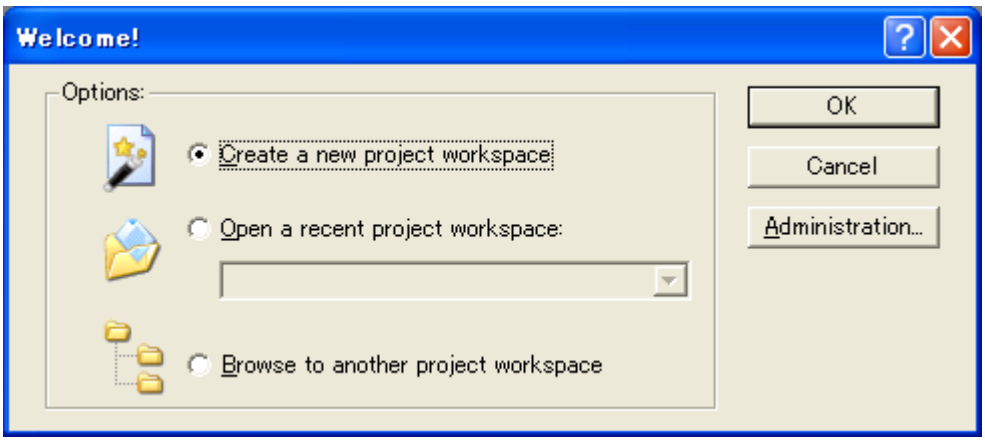


Figure 4.6 [Welcome!] Dialog Box

2. The Project Generator is started.
If you have purchased the toolchain, the following dialog box is displayed.

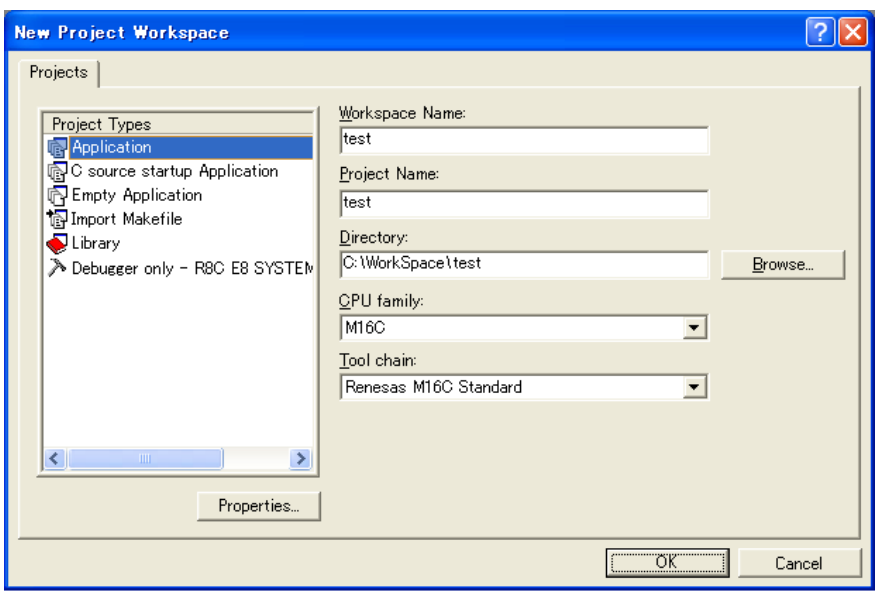


Figure 4.7 [New Project Workspace] Dialog Box

[Workspace Name] edit box:	Enter the new workspace name. Here, enter ‘test’.
[Project Name] edit box:	Enter the project name. When the project name is the same as the workspace name, it needs not be entered.
[CPU family] drop-down list box:	Select the target CPU family.
[Tool chain] drop-down list box:	Select the target toolchain name when using the toolchain. Otherwise, select [None].
[Project type] list box:	Select the project type to be used.

3. Make the required setting for the toolchain.

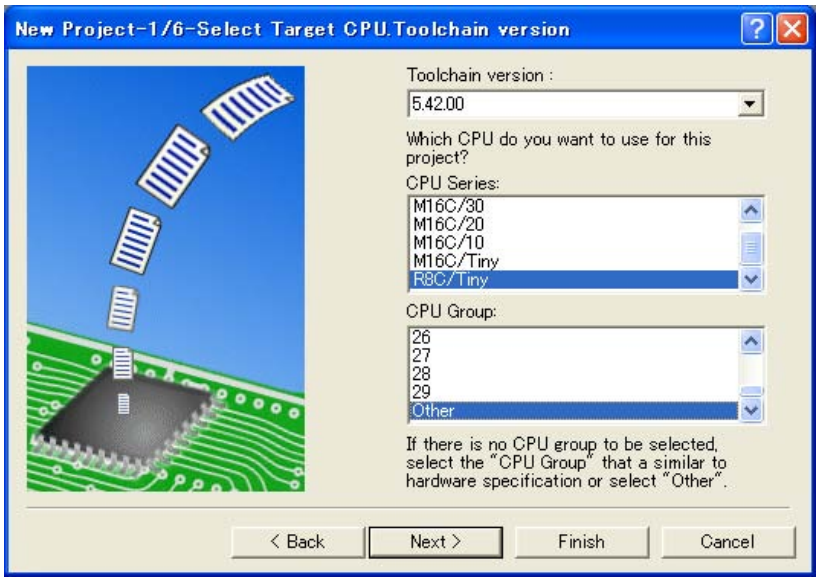


Figure 4.8 [New Project-1/6-Select Target CPU. Toolchain version] Dialog Box

Select your toolchain version, CPU series and click the [Next] button.

4. Make the required settings for the RTOS.

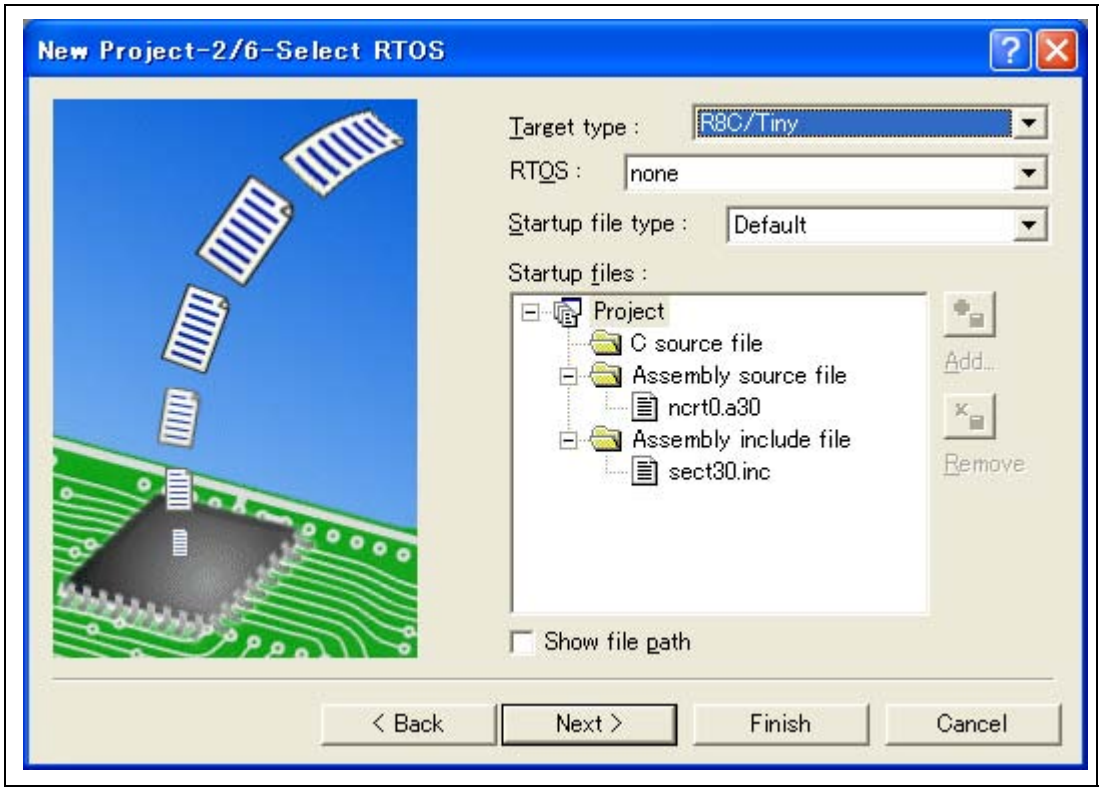


Figure 4.9 [New Project-2/6-Select RTOS] Dialog Box

Select the RTOS you use, type of startup file and click the [Next] button.

Note: In the H8/Tiny, H8/Super Low Power series, and 740 Family, this dialog box will not be displayed.

5. Make the required settings for the heap area etc.

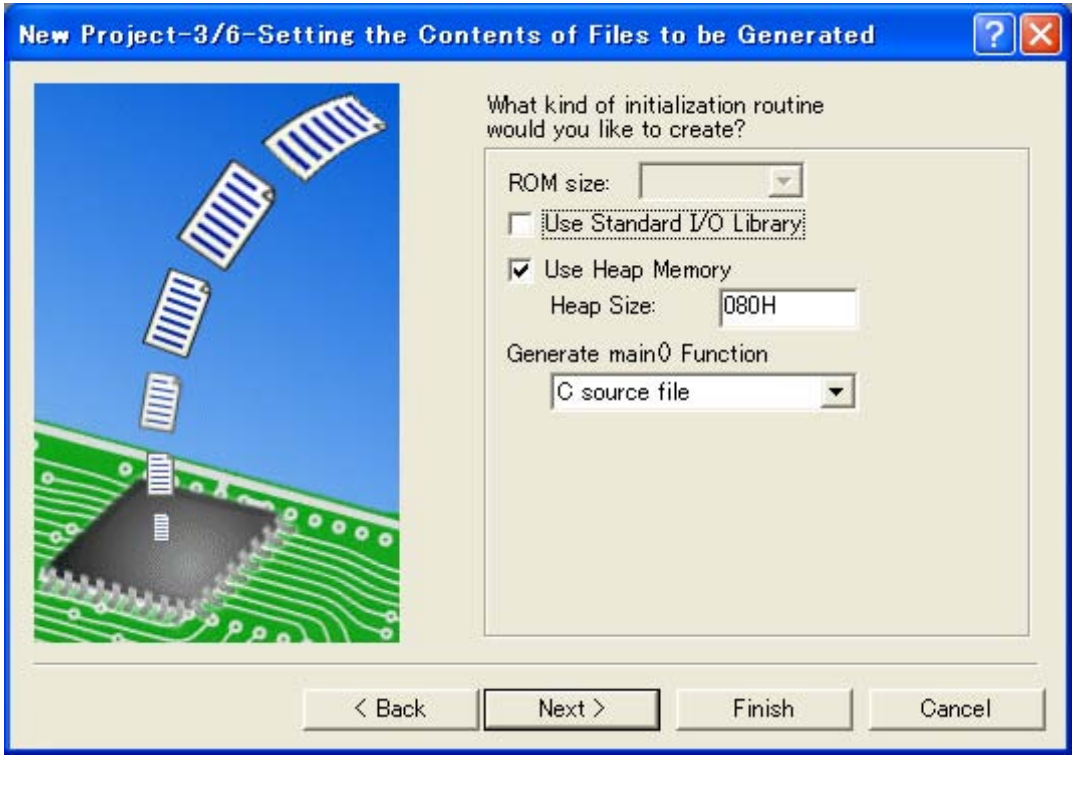


Figure 4.10 [New Project-3/6-Setting the Contents of Files to be Generated] Dialog Box

Set the heap size etc. and click the [Next] button.

Note: In the H8/Tiny, H8/Super Low Power series, and 740 family, the content of the dialog box will differ.

6. Make the required setting for the stack area.

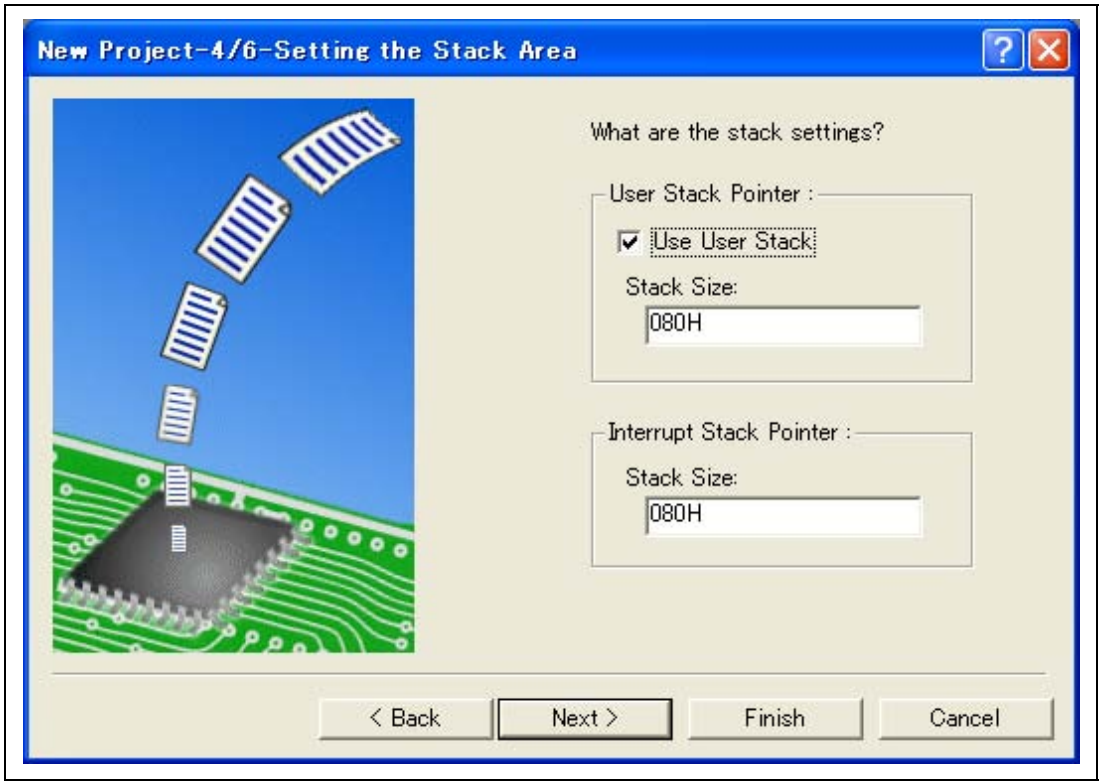


Figure 4.11 [New Project-4/6-Setting the Stack Area] Dialog Box

Set the stack size and click the [Next] button.

7. When the tool chain settings have been completed, the following dialog box is displayed.

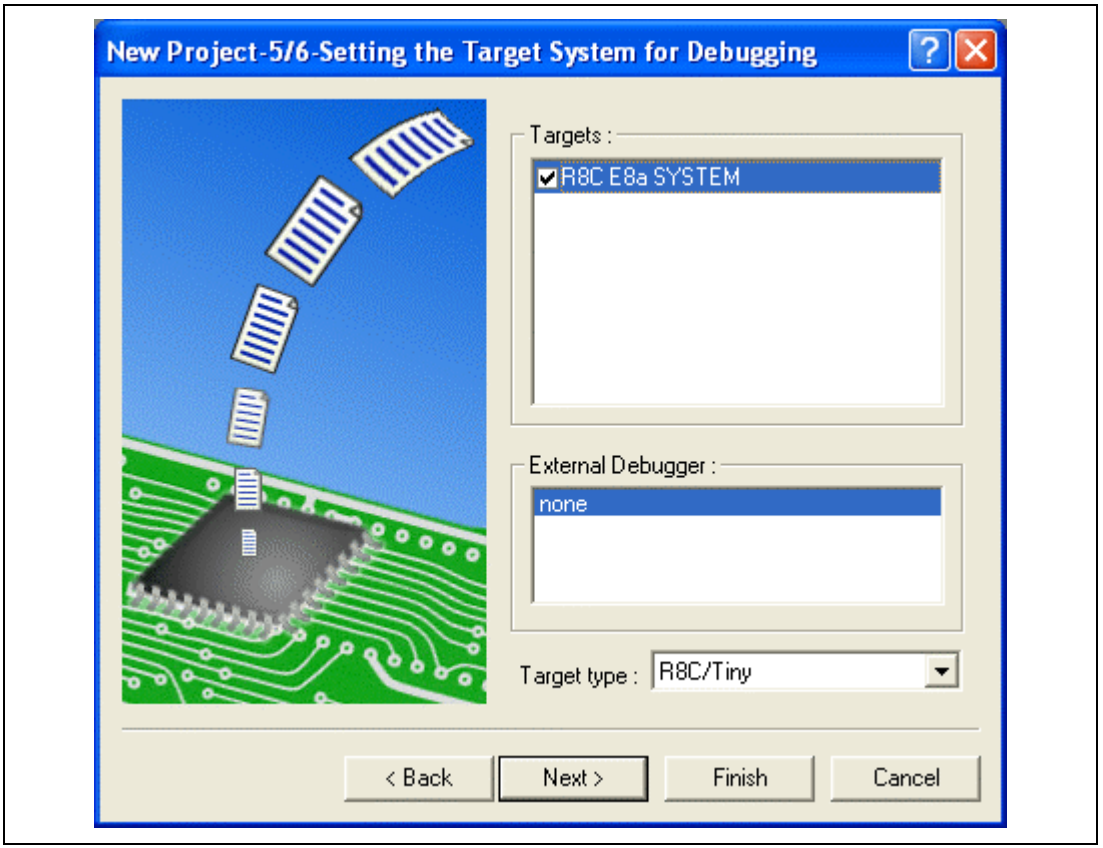


Figure 4.12 [New Project-5/6-Setting the Target System for Debugging] Dialog Box

Select the target system for debugging and click the [Next] button.

[Target system for debugging]	[Target MCU]
M32C E8a SYSTEM:	M32C
M16C E8a SYSTEM:	M16C
R8C E8a SYSTEM:	R8C/Tiny
740 E8a SYSTEM:	740
H8 Tiny/Super Low Power E8a SYSTEM 300H:	H8/300H Tiny, H8/300H Super Low Power
H8 Tiny/Super Low Power E8a SYSTEM 300L:	H8/300 Super Low Power

Select another product, if necessary.

8. Set a configuration file name. The configuration file saves the settings of the High-performance Embedded Workshop except for the emulator.

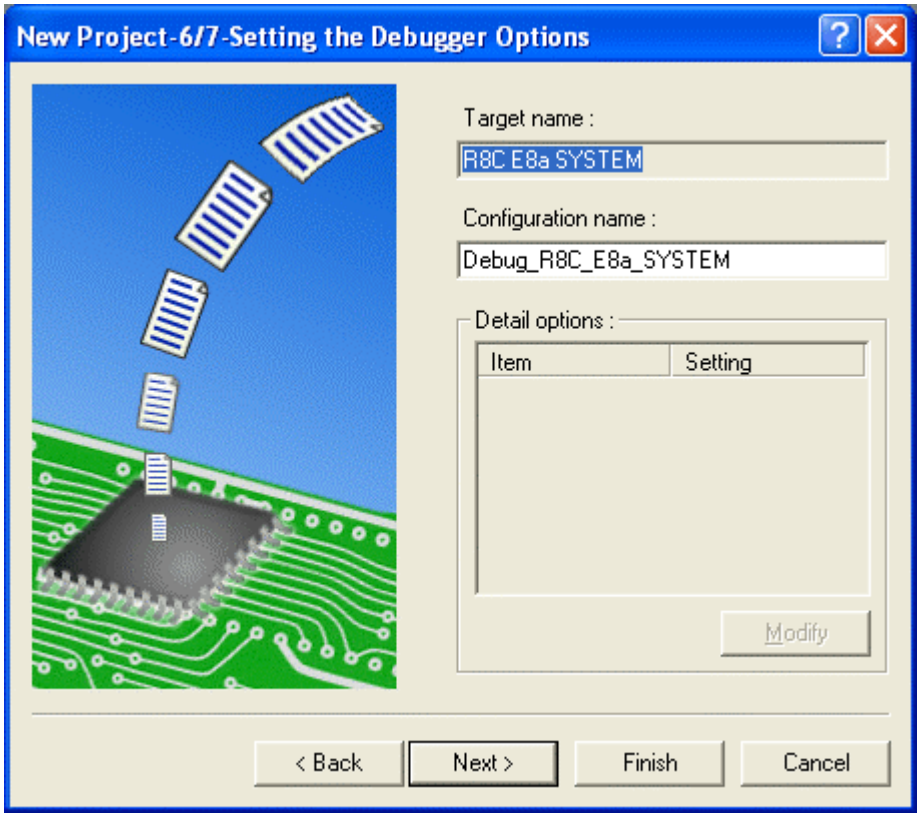


Figure 4.13 [New Project-6/7-Setting the Debugger Options] Dialog Box

9. Finally, confirm the file name you create.

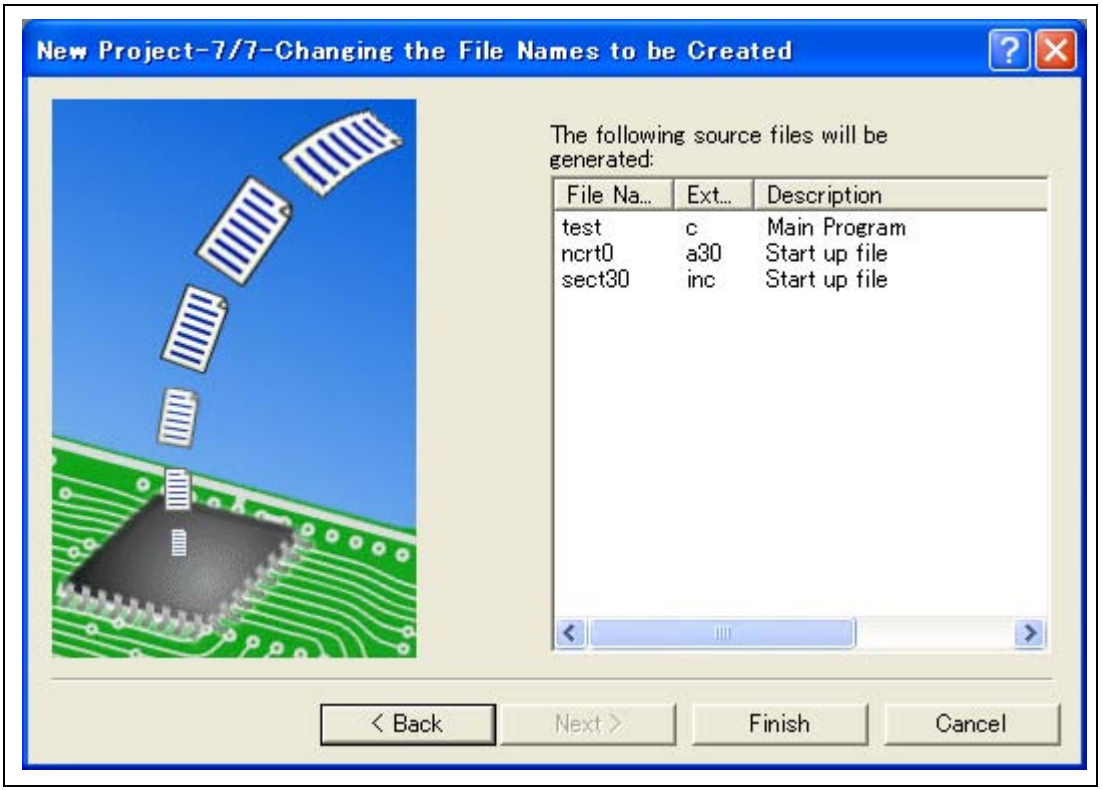


Figure 4.14 [New Project-7/7-Changing the File Names to be Created] Dialog Box

The files which will be generated by the High-performance Embedded Workshop are displayed
If you want to change the file name, select and click it then enter the new name.

This is the end of the emulator settings.

Exit the Project Generator following the instructions on the screen. The High-performance Embedded Workshop is activated.

10. After the High-performance Embedded Workshop has been activated, connect the emulator. However, it is not needed to connect the emulator immediately after the High-performance Embedded Workshop has been activated.
    To connect the emulator, use one of the methods (a) and (b) below. For operation during connection, refer to section 3.8, System Check.

    (a) Connecting the emulator after the setting at emulator activation

    Select [Debug settings…] from the [Debug] menu to open the [Debug Settings] dialog box. It is possible to register the download module or the command chain that is automatically executed at activation. For details on the [Debug Settings] dialog box, refer to section 4.2, Setting at E8a Emulator Activation.

    After the [Debug Settings] dialog box has been set, when the dialog box is closed, the emulator is connected.

(b) Connecting the emulator without the setting at emulator activation

The emulator can be easily connected by switching the session file that the setting for the emulator use has been registered.

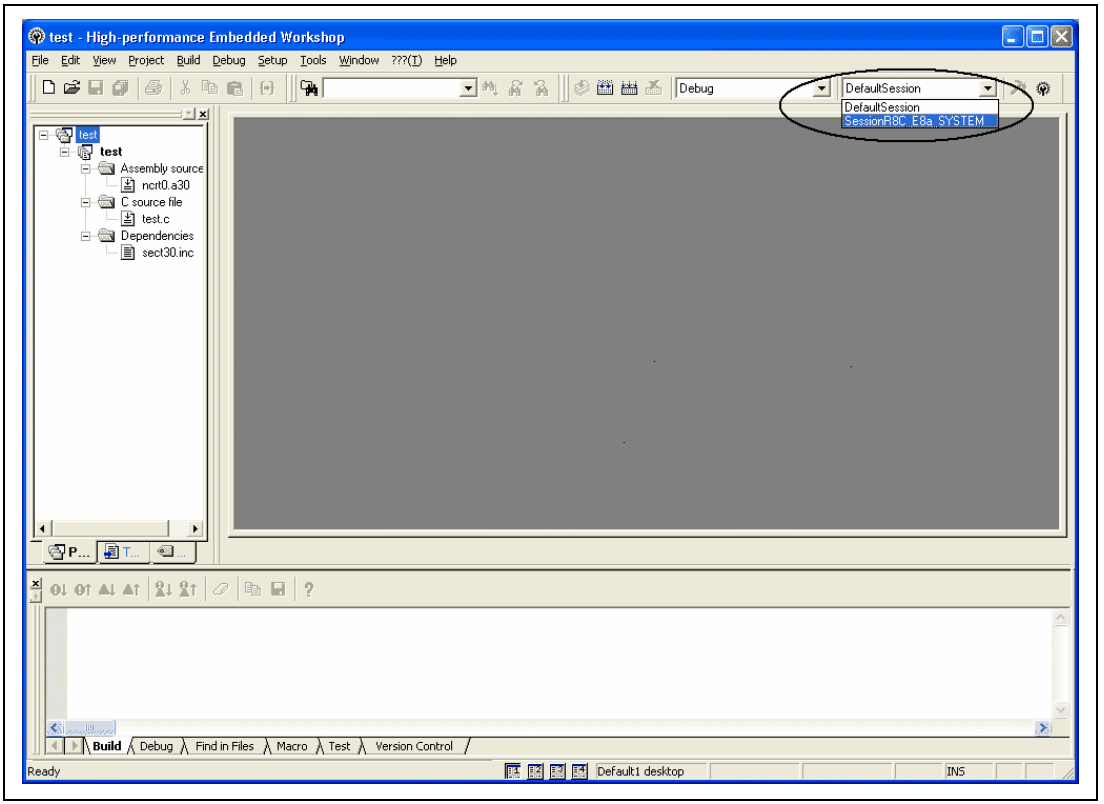


Figure 4.15 Selecting the Session File

In the list box that is circled in Figure 4.15, select the session file name including the character string that has been set in the [Target name] text box in Figure 4.13, [New Project-6/7-Setting the Debugger Options] dialog box. The setting for using the emulator has been registered in this session file.

After selected, the emulator is automatically connected.

4.1.3 Selecting an Existing Workspace

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select [Browse to another project workspace] radio button and click the [OK] button.

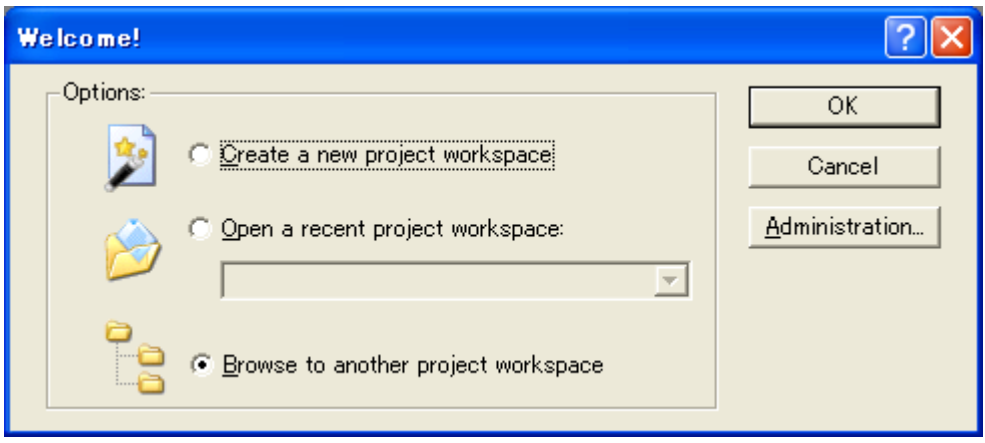


Figure 4.16 [Welcome!] Dialog Box

2. The [Open Workspace] dialog box is displayed. Select a directory in which you have created a workspace.
   After that, select the workspace file (.hws) and press the [Open] button.

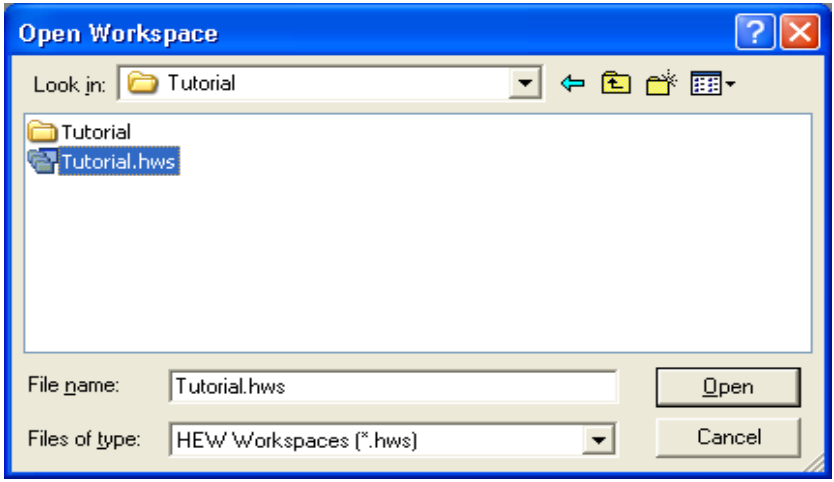


Figure 4.17 [Open Workspace] Dialog Box

3. This activates the High-performance Embedded Workshop and recovers the state of the selected workspace at the time it was saved.
   When the saved state information of the selected workspace includes connection to the emulator, the emulator will automatically be connected. To connect the emulator when the saved state information does not include connection to the emulator, refer to section 4.1.2, Creating the New Workspace (Toolchain Used).

4.2 Setting at E8a Emulator Activation

4.2.1 Setting at Emulator Activation

When the emulator is activated, the command chain can be automatically executed. It is also possible to register multiple load modules to be downloaded. The registered load modules are displayed on the workspace window.

1. Select [Debug Settings…] from the [Debug] menu to open the [Debug Settings] dialog box.

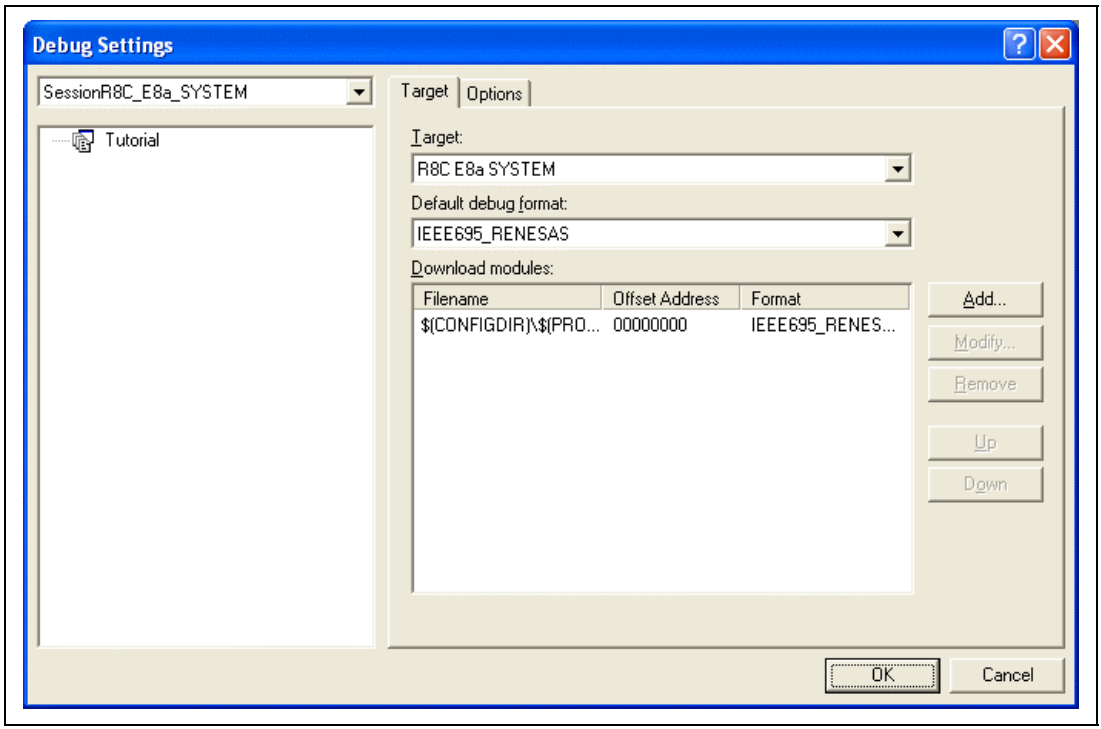


Figure 4.18 [Debug Settings] Dialog Box ([Target] Page)

2. Select the product name to be connected in the [Target] drop-down list box.
3. Select the format of the load module to be downloaded in the [Default Debug Format] drop-down list box, then register the corresponding download module in the [Download Modules] list box.
4. Click the [Options] tab.

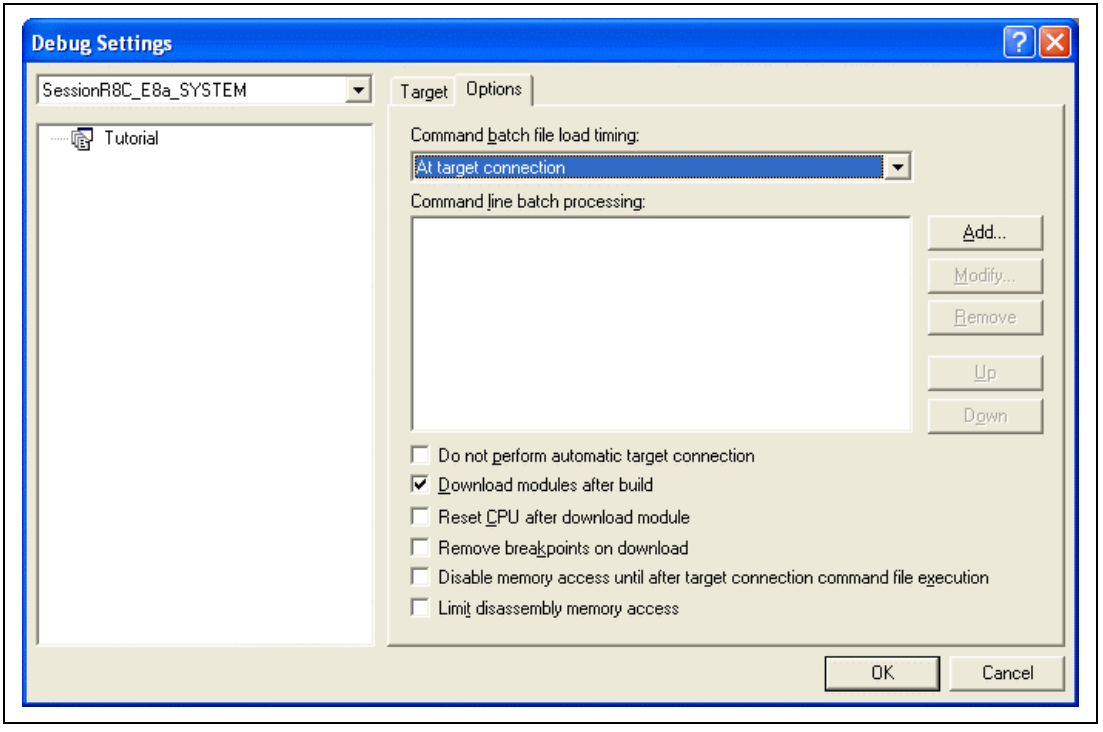


Figure 4.19 [Debug Settings] Dialog Box ([Options] Page)

The command chain that is automatically executed at the specified timing is registered. The following three timings can be specified:

- At connecting the emulator
- Immediately before downloading
- Immediately after downloading

Specify the timing for executing the command chain in the [Command batch file load timing] drop-down list box. In addition, register the command-chain file that is executed at the specified timing in the [Command Line Batch Processing] list box.

4.2.2 Downloading a Program

A download module is added under [Download modules] in the [Workspace] window.

Open the load module of [Download modules] in the [Workspace] window by clicking the right-hand mouse button and select [Download module] to start downloading the module.

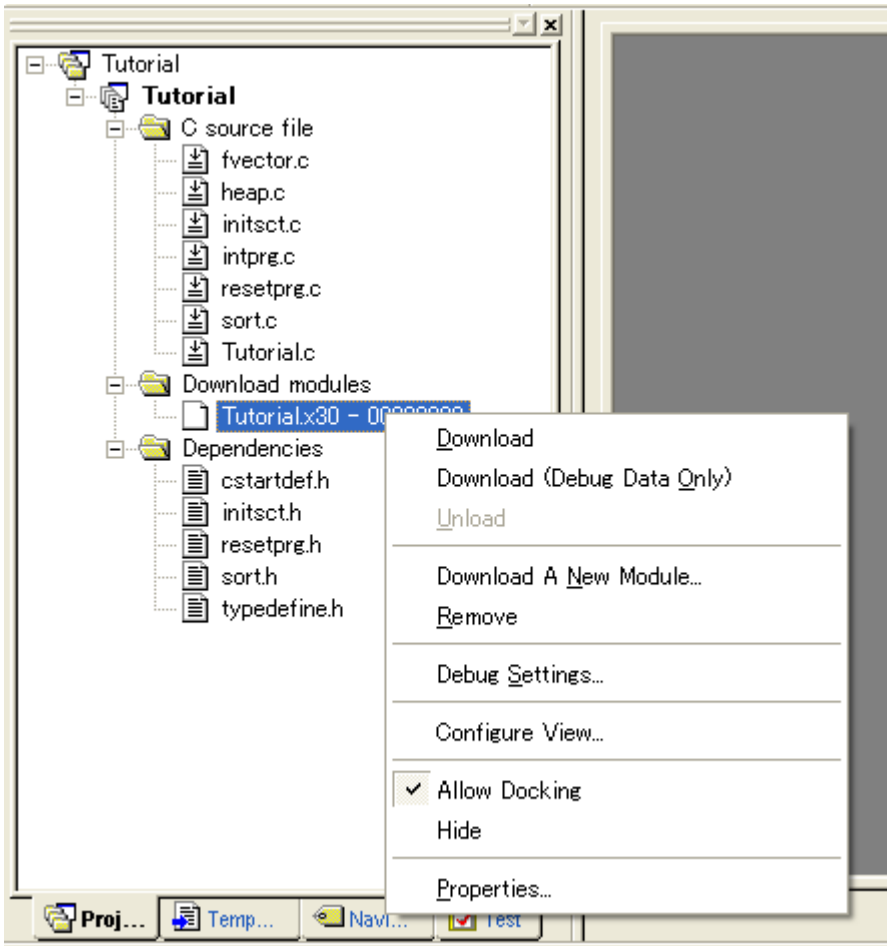


Figure 4.20 Download Menu of the [Workspace] Window ([Projects])

Notes: 1. When load modules are downloaded, select [Debug] -> [Download] -> [All Download modules].

2. The emulator downloads programs to the flash memory just before execution of the user program.

4.2.3 Setting the [Program Flash] Mode

The following describes the procedures when the emulator is used as the programming tool. The load module to be downloaded to the new workspace is registered and programmed.

(a) Select the new project workspace.

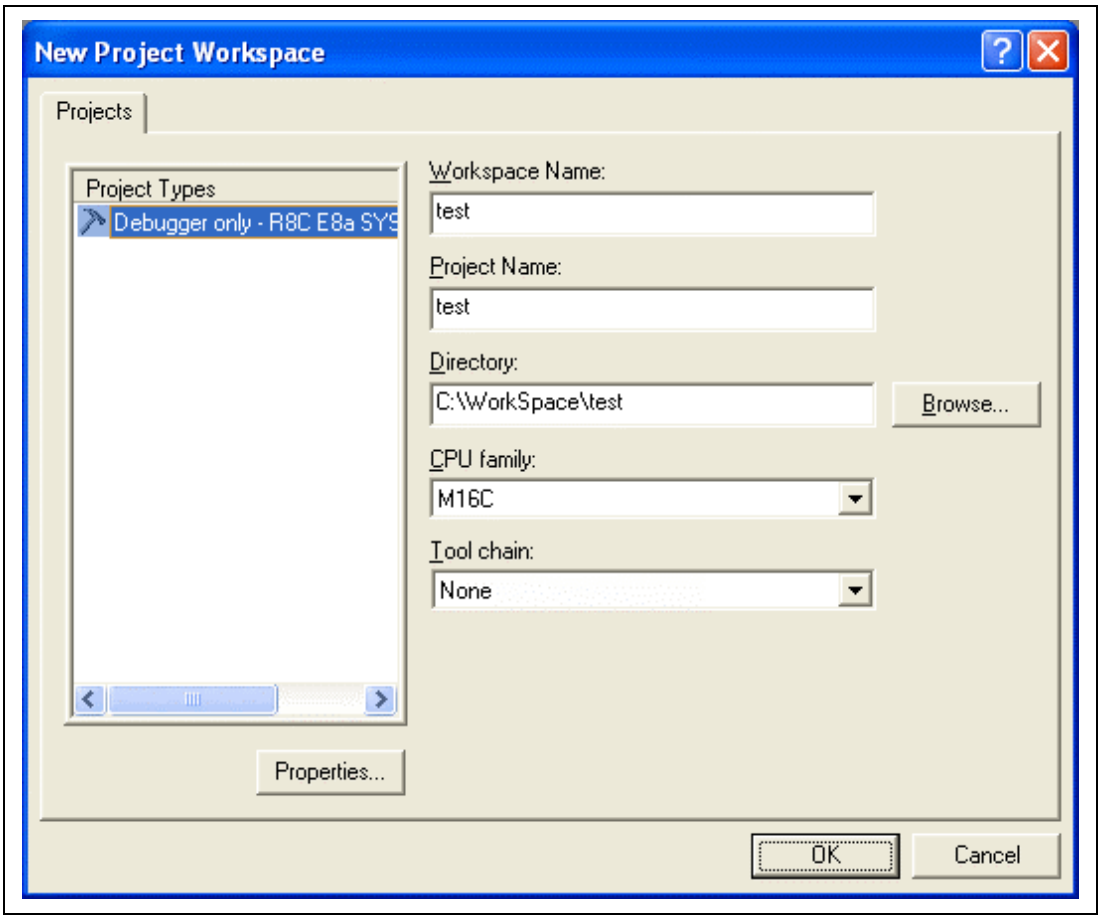


Figure 4.21 [New Project Workspace] Dialog Box

(b) Select the target MCU and click the [Next] button.

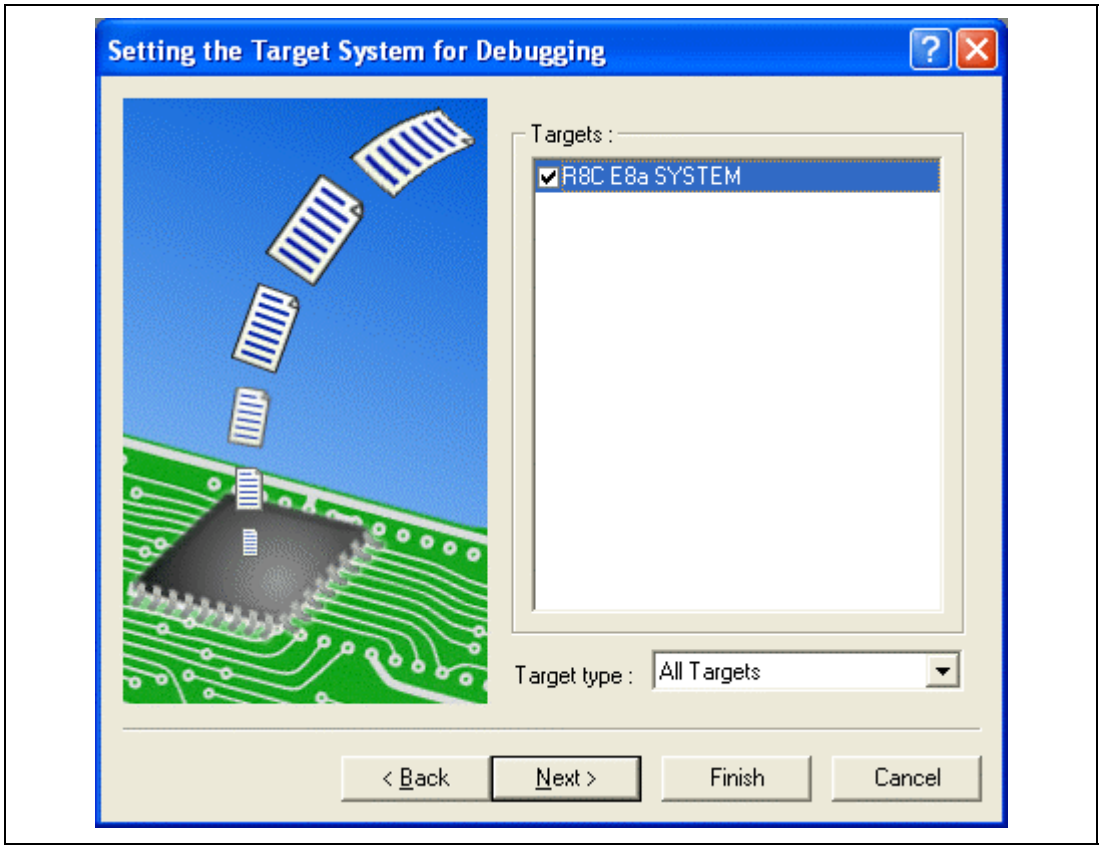


Figure 4.22 [Setting the Target System for Debugging] Dialog Box

(c) Set a configuration file name and click the [Finish] button.

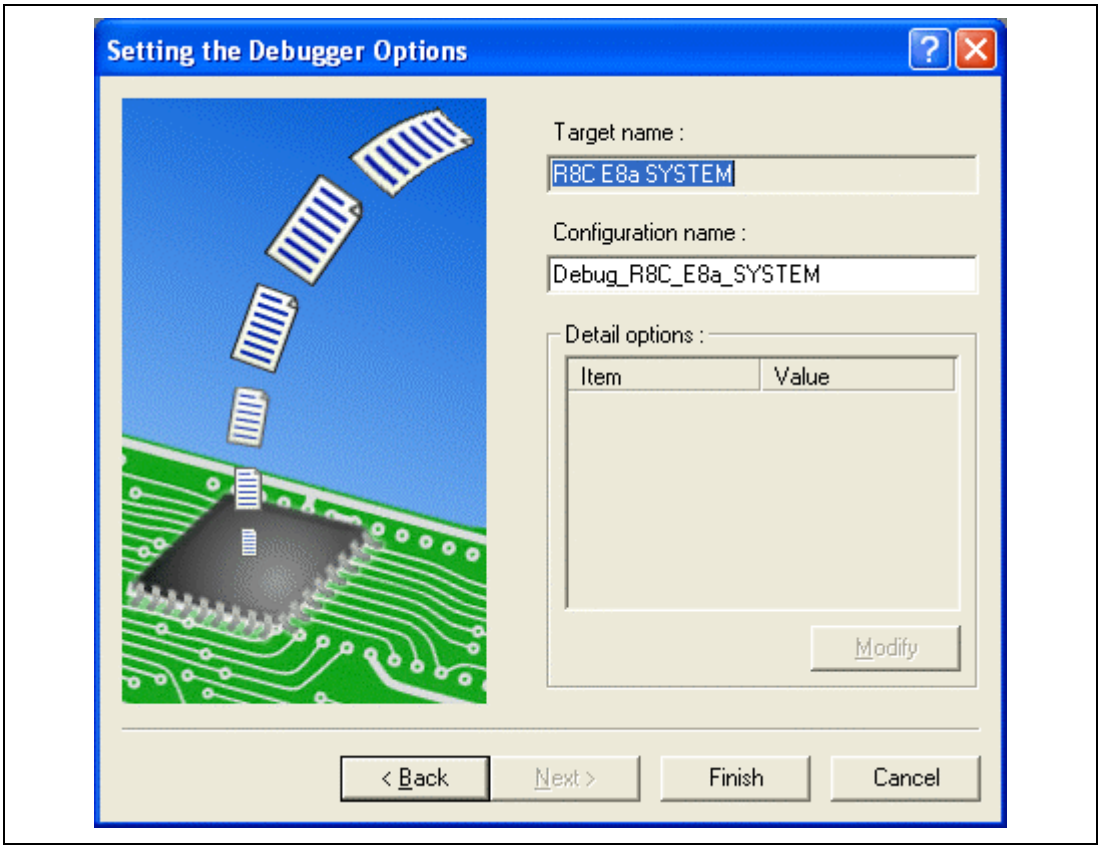


Figure 4.23 [Setting he Debugger Options] Dialog Box

(d) The [Emulator Setting] dialog box is displayed.

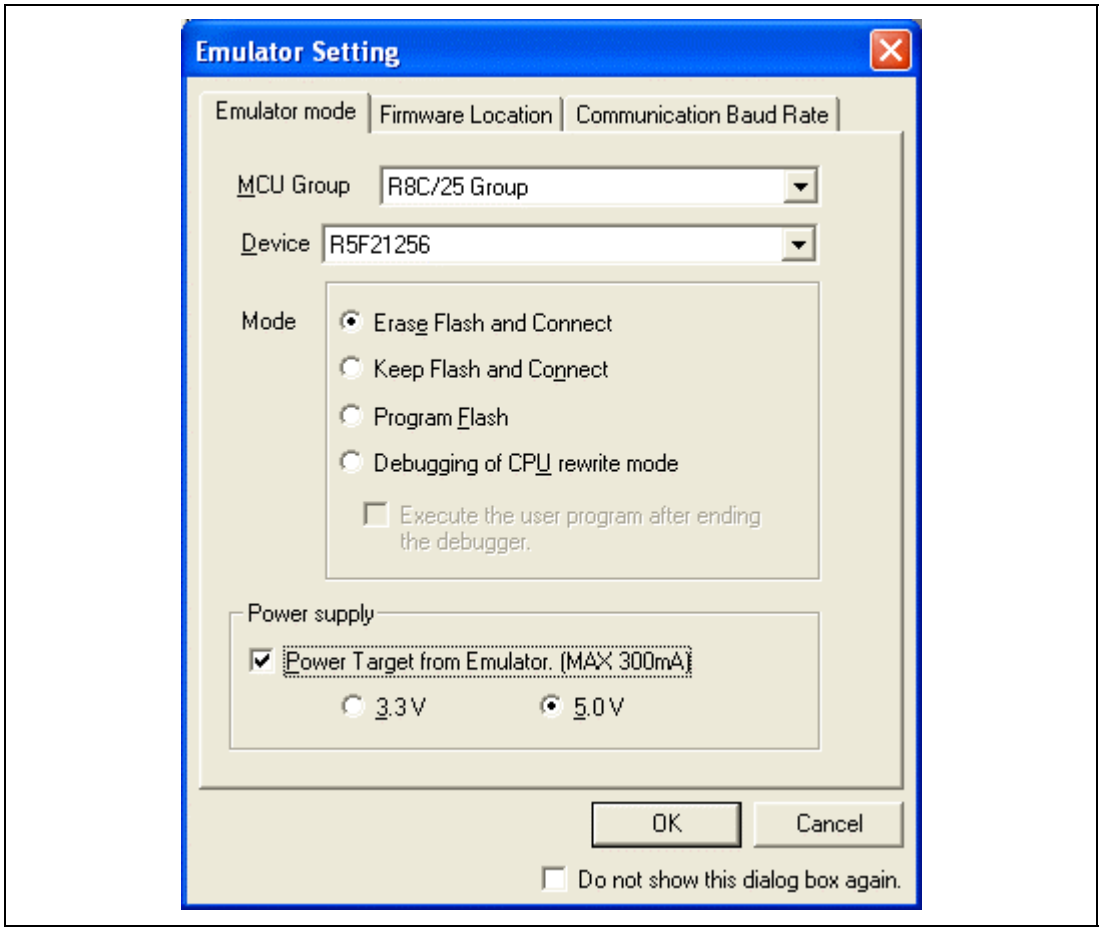


Figure 4.24 [Emulator Setting] Dialog Box

Select the [Program Flash] mode.

When the [Power Target from Emulator. (MAX 300mA)] check box is selected, power will be supplied to the user system up to 300 mA. Then select 3.3 V or 5.0 V according to the power voltage of the user system.

(e) When [Power Target from Emulator. (MAX 300mA)] is not checked, the dialog box shown in Figure 4.25 will be displayed.

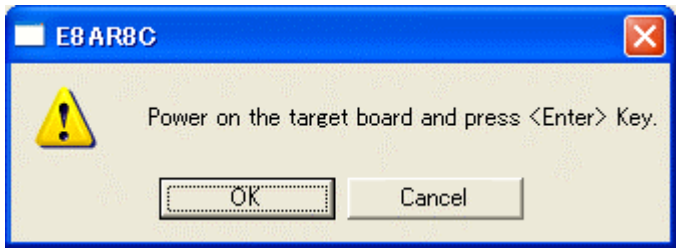


Figure 4.25 Dialog Box of Power-on request message

When [Power Target from Emulator. (MAX 300mA)] is not checked, turn on the user system.

(f) Enter the ID code printed on the flash memory.

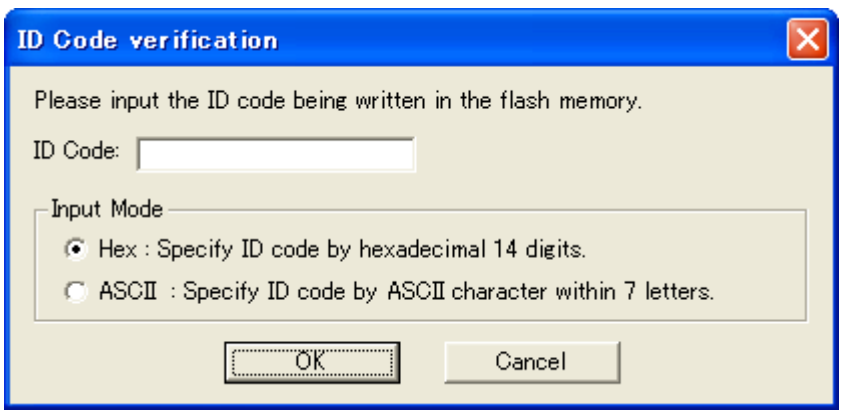

ID Code verification
Please input the ID code being written in the flash memory.
ID Code:
Input Mode
Hex : Specify ID code by hexadecimal 14 digits.
ASCII : Specify ID code by ASCII character within 7 letters.
OK
Cancel

Figure 4.26 [ID Code verification] Dialog Box

Note: Opening the [ID Code verification] dialog box in [Program Flash] mode depends on the target device. For details, refer to the additional document, Notes on Connecting the xxxxx.

(g) Select [Debug Settings…] from the [Debug] menu.

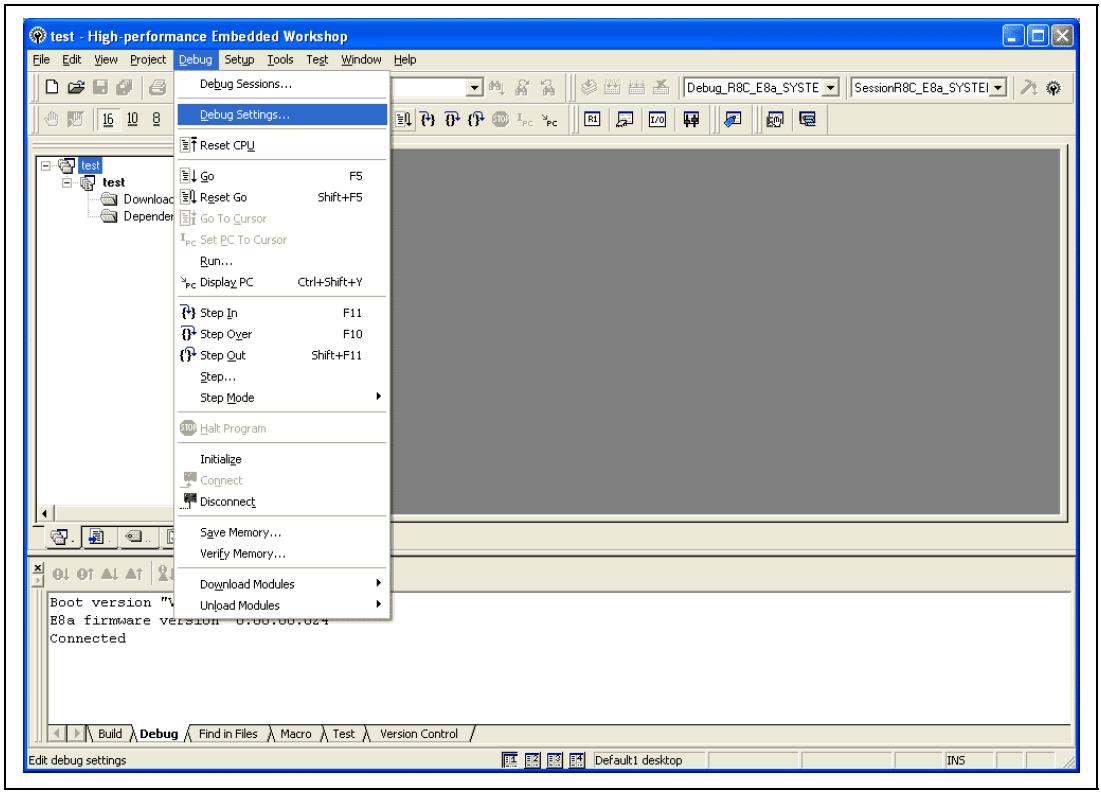


Figure 4.27 High-performance Embedded Workshop Window

(h) Select the target MCU and then the download module with the [Add…] button.

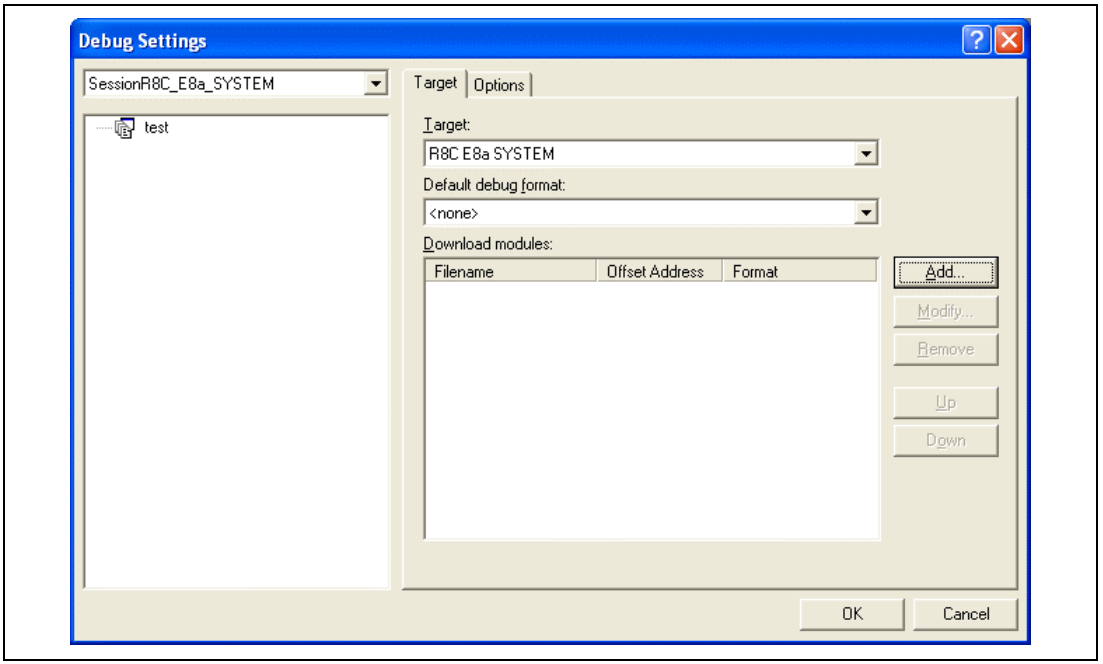


Figure 4.28 [Debug Settings] Dialog Box ([Target] Page)

(i) The download file is displayed on [Projects].

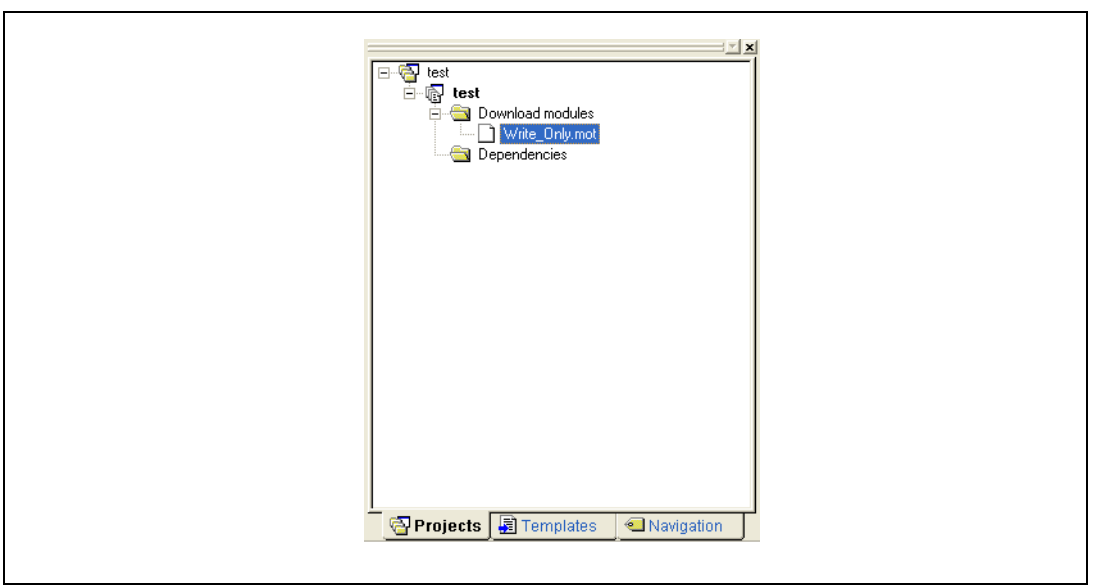


Figure 4.29 [Workspace] Window ([Projects])

(j) Select and download the file with the right-hand mouse button.

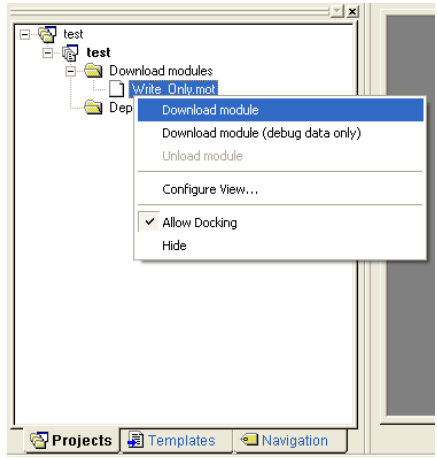


Figure 4.30 Download Menu of the [Workspace] Window ([Projects])

(k) The dialog box for sum checking is displayed and writing is completed.

Figure 4.31 Message for Completion of Flash Memory Writing

(l) When the following dialog box is displayed, close and restart or exit the workspace.

Figure 4.32 Message for Restarting or Exiting the [Program Flash] Mode

Section 5 Debugging

This section describes the debugging operations and their related windows and dialog boxes. For details on the common functions of the High-performance Embedded Workshop products, refer to the High-performance Embedded Workshop User's Manual.

5.1 Setting the Environment for Emulation

5.1.1 Opening the [Configuration] Dialog Box

Selecting [Setup -> Emulator -> System…] or clicking the [Emulator System] toolbar button ( ) opens the [Configuration] dialog box.

5.1.2 [General] Page

Sets the emulator operation conditions.

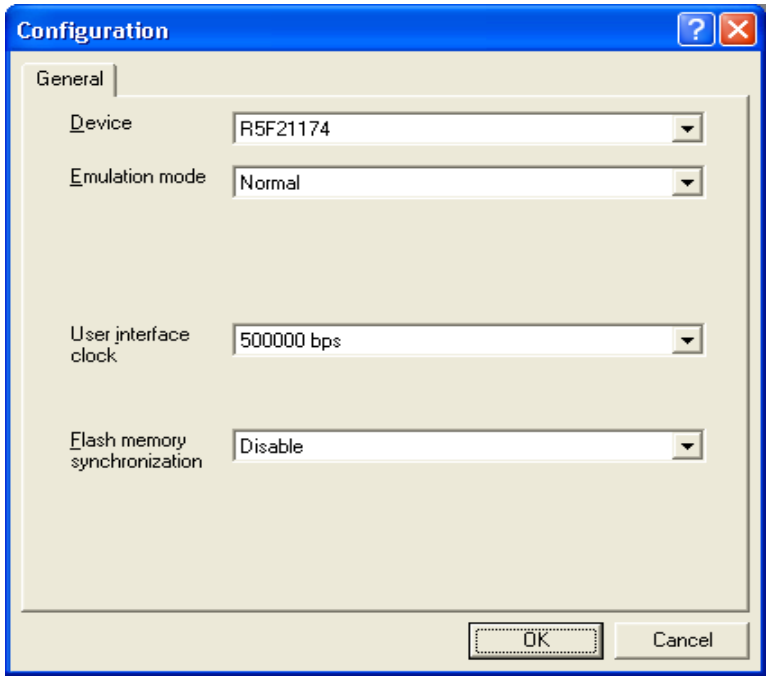


Figure 5.1 [Configuration] Dialog Box ([General] Page)

Items that can be displayed in the sheet are listed below.

[Device] combo box	Displays the MCU name.
[Emulation mode] combo box	Selects the emulation mode at user program execution. Select Normal to perform normal emulation. Select No break to disable PC breakpoint or break condition settings during emulation.
[User interface clock] combo box	Displays the transfer clock of the user interface.

[Flash memory synchronization] combo box	Selects whether or not the contents of the flash memory are acquired by the emulator when the user program is stopped or the position where the PC break is set is put back as the original code. When the flash memory is not programmed by the user program, its contents need not be acquired by the emulator. If there is no problem with the state that the program in the flash memory has been replaced as the PC break code, the position where the PC break is set needs not be put back as the original code. Disable: Read or program is not performed for the flash memory except when the emulator is activated, the flash memory area is modified, and the settings of the PC break to the flash memory area are changed. PC to flash memory: When the user program is stopped, the specified PC break code is replaced as the original instruction. Select this option if there is a problem with the state that the program in the flash memory has been replaced as the PC break code. Flash memory to PC: When the user program is stopped, the contents of the flash memory are read by the emulator. Select this option if the flash memory is reprogrammed by the user program. PC to flash memory, Flash memory to PC: When the user program is stopped, the contents of the flash memory are read by the emulator and the specified PC break code is replaced as the original instruction. Select this option if the flash memory is reprogrammed by the user program and there is a problem with the state that the program in the flash memory has been replaced as the PC break code.

Note: This dialog box differs depending on products. For details, refer to the online help.

5.2 Downloading a Program

This section describes how to download a program and view it as source code or assembly-language mnemonics.

Note: After a break has been detected, the High-performance Embedded Workshop displays the location of the program counter (PC). In most cases, for example if an Elf/Dwarf2-based project is moved from its original path (at the build), the source file may not be automatically found. In this case, the High-performance Embedded Workshop will open a source file browser dialog box to allow you to manually locate the file.

5.2.1 Downloading a Program

A load module to be debugged must be downloaded.

To download a program, select the load module from [Debug -> Download] or select [Download] from the popup menu opened by clicking the right-hand mouse button on the load module in [Download modules] of the [Workspace] window.

Note: Before downloading a program, it must be registered to the High-performance Embedded Workshop as a load module. For registration, refer to section 4.2, Setting at E8a Emulator Activation.

5.2.2 Viewing the Source Code

Select a source file name in the [Workspace] window and click [Open] in the popup menu to make the High-performance Embedded Workshop display the source file in the editor. It is also possible to display your source files by double-clicking on their names in the [Workspace] window.

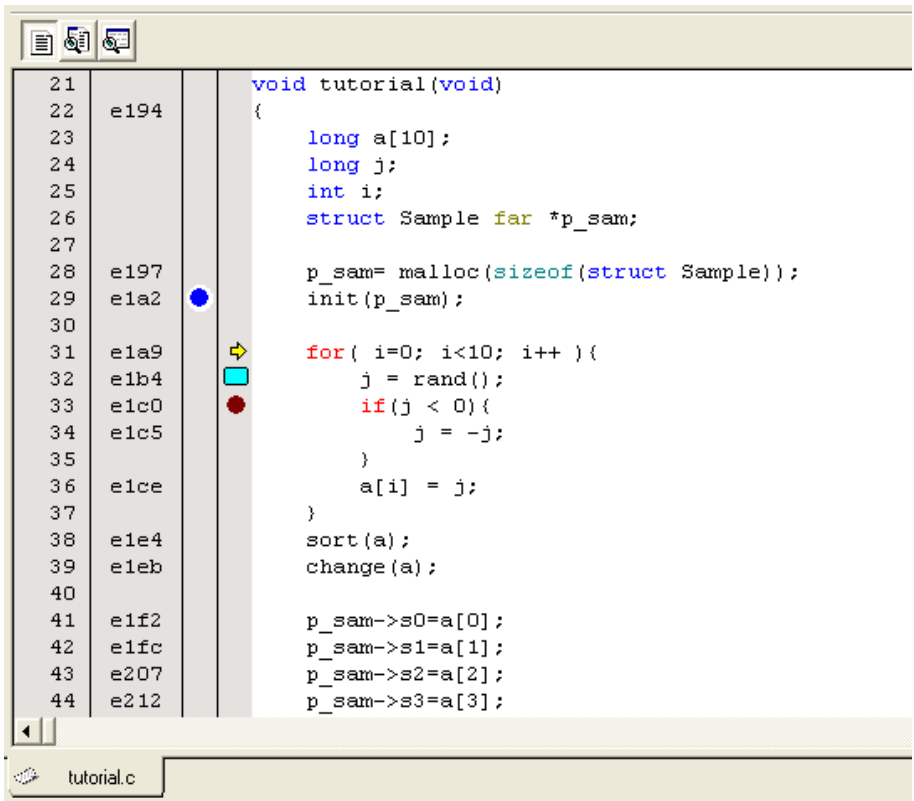


Figure 5.2 [Source] Window

In this window, the following items are shown on the left as line information.

The first column (Line Number column) Line number

The second column (Source address column): Address information

The third column (Event column): Event information (break condition)

The fourth column (S/W Breakpoints column): PC, bookmark, and breakpoint information

The right window that shows the source is called as the [Source] window.

- Line Number column

A line number for the current source file is displayed.

- Source address column

When a program is downloaded, an address for the current source file is displayed on the Source address column. These addresses are helpful when setting the PC value or a breakpoint.

Event column

The Event column displays the following item:

: An address condition as a break condition is set.

: An address interrupt break is set.

This is also set by using the popup menu.

The bitmap symbol above is shown by double-clicking the Event column. This is also set by using the popup menu.

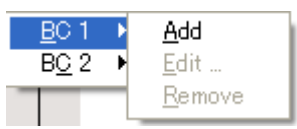


Figure 5.3 Popup Menu

S/W Breakpoints column

S/W Breakpoints column displays the following items:

: A bookmark is set.

: A PC Break is set.

: PC location

➲ To switch off a column in all source files

1. Click the right-hand mouse button on the [Source] window or select the [Edit] menu.
2. Click the [Define Column Format…] menu item.
3. The [Global Editor Column States] dialog box is displayed.
4. A check box indicates whether the column is enabled or not. If it is checked, the column is enabled. If the check box is gray, the column is enabled in some files and disabled in others. Deselect the check box of a column you want to switch off.
5. Click the [OK] button for the new column settings to take effect.

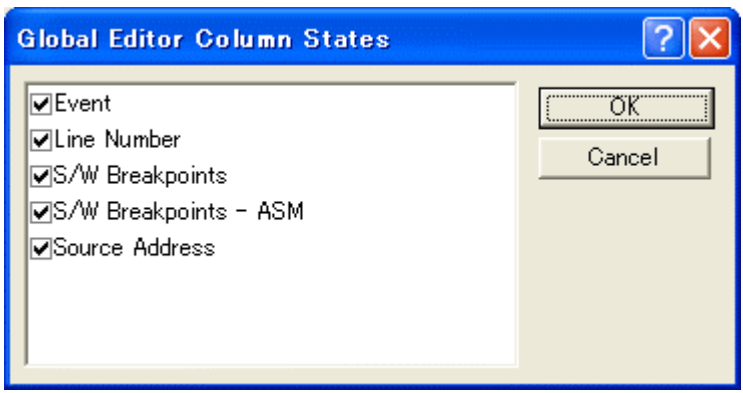


Figure 5.4 [Global Editor Column States] Dialog Box

➲ To switch off a column in one source file

1. Open the source file which contains the column you want to remove and click the [Edit] menu.
2. Click the [Columns] menu item to display a cascaded menu item. The columns are displayed in this popup menu. If a column is enabled, it has a tick mark next to its name. Clicking the entry will toggle whether the column is displayed or not.

5.3 Stopping Your Program

This section describes how you can halt execution of your application's code. This section describes how to stop the program by setting an address match breakpoints at specific locations in your code.

5.3.1 Address Match Breakpoints

When you are trying to debug your program you will want to be able to stop the program running when it reaches a specific point or points in your code. You can do this by setting the address match breakpoint on the line or instruction at which you want the execution to stop. The following instructions will show you how to quickly set and clear address match breakpoint.

- To set a PC breakpoint in the [Source] window

1. Make sure that the [Disassembly] or [Source] window is open at the place you want to set a PC breakpoint.
2. Double-click in the event column at the line which you want the program to stop.
3 You will see a blue circle appear in the column to indicate that an address match breakpoint has been set.

Now when you run your program and it reaches the address at which you set the address match breakpoint, execution halts with the message "BREAK CONDITION" displayed in the [Debug] sheet of the [Output] window, and the [Source] or [Disassembly] window is updated with the address match breakpoint line marked with an arrow in the gutter.

Note: When a break occurs, the program stops just before it is about to execute the line or instruction at which you set an address match breakpoint. If you choose Go or Step after stopping at the address match breakpoint, then the line marked with an arrow will be the next instruction to be executed.

About address match breakpoints, see also section 5.4.13, Setting Break Conditions.

5.4 Using the Event Points

The emulator has the event point function that performs breaking by specifying higher-level conditions along with the PC breakpoints standard for the High-performance Embedded Workshop.

5.4.1 PC Breakpoints

When the instruction of the specified address is fetched, the user program is stopped. Up to 255 points can be set.

5.4.2 Break Conditions

Break conditions can be used for higher-level conditions such as the data condition as well as specification of the single address.

5.4.3 Opening the [Event] Window

Select [View -> Code -> Eventpoints] or click the [Eventpoints] toolbar button ( ) to open the [Event] window.

The [Event] window has the following two sheets:

[Breakpoint] sheet: Displays the settings made for PC breakpoints. It is also possible to set, modify, and cancel PC breakpoints.

[Breakcondition] sheet: Displays or sets the settings made for break condition channels.

5.4.4 Setting PC Breakpoints

It is possible to display, modify, and add PC breakpoints on the [Breakpoint] sheet.

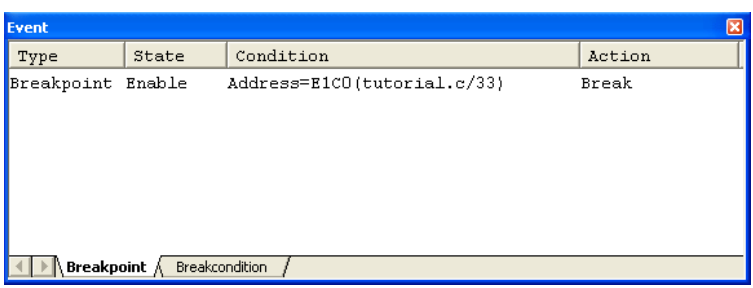


Figure 5.5 [Event] Window ([Breakpoint] Sheet)

This window displays and sets the breakpoints. Items that can be displayed in the sheet are listed below.

[Type] Breakpoint

[State] Whether the breakpoint is enabled or disabled

[Condition] An address that the breakpoint is set
Address = Program counter (Corresponding file name, line, and symbol name)

[Action] Operation of the emulator when a break condition is satisfied
Break: Halts execution

When a breakpoint is double-clicked in this window, the [**Set Break**] dialog box is opened and break conditions can be modified.

A popup menu containing the following options is available by right-clicking within the window.

5.4.5 Add

Sets breakpoints. Clicking this item will open the [**Set Break**] dialog box and break conditions can be specified.

5.4.6 Edit

Only enabled when one breakpoint is selected. Select a breakpoint to be edited and click this item. The [**Set Break**] dialog box will open and break conditions can be changed.

5.4.7 Enable

Enables the selected breakpoint(s).

5.4.8 Disable

Disables the selected breakpoint(s). When a breakpoint is disabled, the breakpoint will remain in the list; when specified conditions have been satisfied, a break will not occur.

5.4.9 Delete

Removes the selected breakpoint. To retain the details of the breakpoint but not have it cause a break when its conditions are met, use the Disable option (see section 5.4.8, Disable).

5.4.10 Delete All

Removes all breakpoints.

5.4.11 Go to Source

Only enabled when one breakpoint is selected. Opens the [**Source**] window at the address of the breakpoint.

5.4.12 [Set Break] Dialog Box

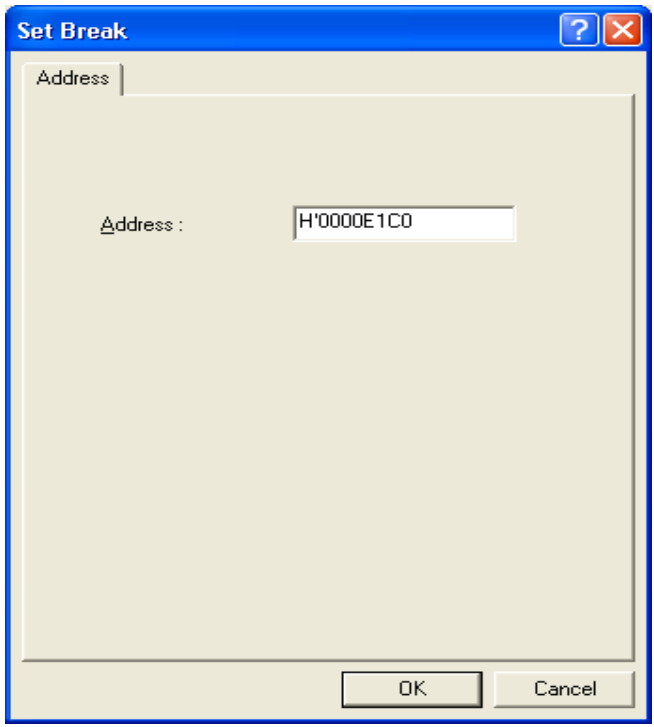


Figure 5.6 [Set Break] Dialog Box

This dialog box specifies break conditions.

A breakpoint address to be set is specified in the **[Address]** edit box. Up to 255 breakpoints can be specified.

When **[Address]** is selected, if an overloaded function or class name including a member function is specified in address, the [**Select Function**] dialog box opens.

Clicking the **[OK]** button sets the break conditions. Clicking the **[Cancel]** button closes this dialog box without setting the break conditions.

Notes on Setting the [Set Break] Dialog Box

1. When an odd address is set for the H8/Tiny and H8/Super Low Power series, the address is rounded down to an even address.
2. A Breakpoint is accomplished by replacing instructions. Accordingly, it can be set only to the flash memory or RAM area. However, a software break cannot be set to the following addresses:
   - An area other than the flash memory or RAM
   - An area occupied by the E8a emulator program
   - An instruction in which Break Condition 1 is satisfied
3. During step execution, a software breakpoint is disabled.
4. A condition set at Break Condition 1 is disabled immediately after starting execution when an instruction at a software breakpoint is executed. A break does not occur even if a condition of Break Condition 1 is satisfied immediately after starting the execution.
5. When execution resumes from the breakpoint address after the program execution stops at the software breakpoint, single-step execution is performed at the address before execution resumes. Therefore, realtime operation cannot be performed.
6. Settings of software breakpoint and Break Condition 1 are invalid while the STEP OVER function is being used.

5.4.13 Setting Break Conditions

On the [Breakcondition] sheet, the settings for break conditions are displayed, modified, and added. For R8C/Tiny series, on the [Breakcondition] sheet, the settings for address match breakpoint are displayed, modified, and added.

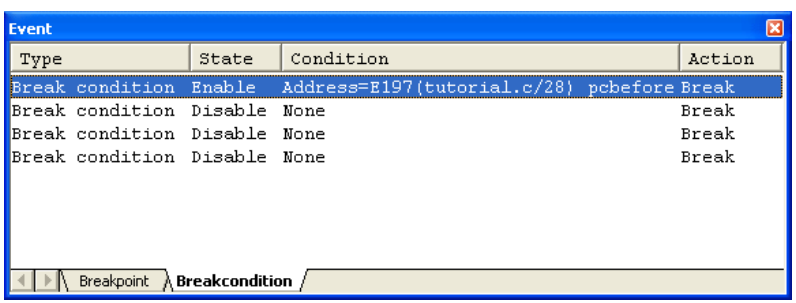


Figure 5.7 [Event] Window ([Breakcondition] Sheet)

Note: The items that can be set in this window differ depending on the product. For the settings for each product, refer to the online help.

This window displays and sets the break condition.

Notes on Setting the Break Condition:

1. When [Go to cursor], [Step In], [Step Over], or [Step Out] is selected, the settings of Break Condition 1 are disabled.
2. Setting of Break Condition 1 is disabled when an instruction to which a BREAKPOINT has been set is executed.
3. When step over function is used, the settings of BREAKPOINT and Break Condition 1 are disabled.

Notes on Setting the Break Condition (for the R8C/Tiny Series):

1. When [Go to cursor], [Step In], [Step Over], or [Step Out] is selected, the settings of Break Condition are disabled.
2. Setting of Break Condition is disabled when an instruction to which a BREAKPOINT has been set is executed.
3. Setting of BREAKPOINT is disabled when BREAKPOINT and Break Condition are set at same address.
4. When step over function is used, the settings of BREAKPOINT and Break Condition are disabled.

Items that can be displayed in the sheet are listed below.

[Type] Break channel number

[State] Whether the breakpoint is enabled or disabled

[Condition] A condition that satisfies a break

[Action] Operation of the emulator when a break condition is satisfied
Break: Halts execution

When a breakpoint is double-clicked in this window, the [**Break condition 1**] dialog box is opened and break conditions can be modified. For details on the [Break condition 1] dialog box, refer to the online help for each product.

A popup menu containing the following options is available by right-clicking within the window.

5.4.14 Edit...

Only enabled when one breakpoint is selected. Select a breakpoint to be edited and click this item. The [**Break condition 1**] dialog box will open and break conditions can be changed.

5.4.15 Enable

Enables the selected break channel(s). A break channel that the condition has not been set is not enabled.

5.4.16 Disable

Disables the selected break channel(s). When a break channel is disabled, a break will not occur even if specified conditions have been satisfied.

5.4.17 Delete

Initializes the condition of the selected break channel. To retain the details of the break channel but not have it cause a break when its conditions are met, use the Disable option (see section 5.4.16, Disable).

5.4.18 Delete All

Initializes conditions of all break channels.

5.4.19 Go to Source

Only enabled when one break channel is selected. Opens the [**Source**] window at address of break channel.

If an address value has not been set to the break channel, this option cannot be used.

5.4.20 Sequential Conditions

Sets the sequential condition of the break channel.

5.4.21 Editing Break Conditions

Handlings for settings other than PC breakpoints and break conditions are common. The following describes examples of such handling.

5.4.22 Modifying Break Conditions

Select a break condition to be modified, and choose [Edit...] from the popup menu to open the dialog box for the event, which allows the user to modify the break conditions. The [Edit...] menu is only available when one break condition is selected.

5.4.23 Enabling Break Conditions

Select a break condition and choose [Enable] from the popup menu to enable the selected break condition.

5.4.24 Disabling Break Conditions

Select a break condition and choose [Disable] from the popup menu to disable the selected break condition. When a break condition is disabled, the break condition will remain in the list, but an event will not occur when the specified conditions have been satisfied.

5.4.25 Deleting Break Conditions

Select a break condition and choose [Delete] from the popup menu to remove the selected break condition. To retain the break condition but not have it cause an event when its conditions are met, use the [Disable] option (see section 5.4.24, Disabling Break Conditions).

5.4.26 Deleting All Break Conditions

Choose [Delete All] from the popup menu to remove all break conditions.

5.4.27 Viewing the Source Line for Break Conditions

Select a break condition and choose [Go to Source] from the popup menu to open the [Source] or [Disassembly] window at the address of the break condition. The [Go to Source] menu is only available when one break condition that has the corresponding source file is selected.

5.4.28 [Break condition x] Dialog Box

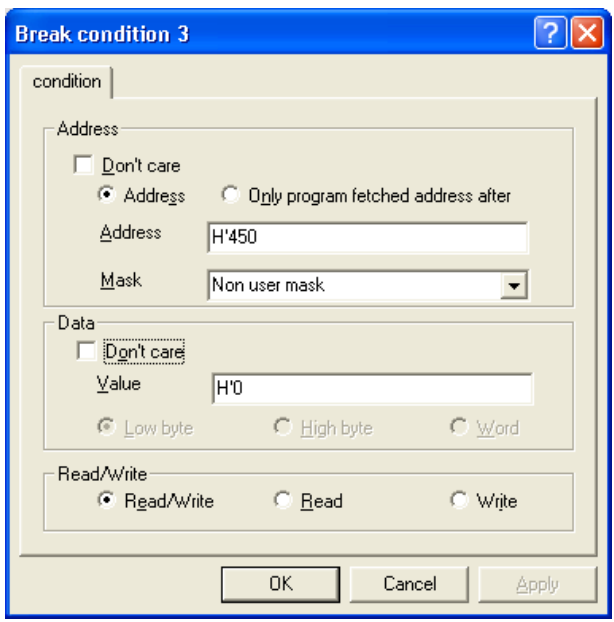


Figure 5.8 [Break condition 3] Dialog Box

This page sets the address bus, data bus, and read/write cycle conditions.

Note: The items that can be set in this dialog box differ depending on the product. For the settings for each product, refer to the online help.

The [Address] group box sets address bus conditions.

Option	Description
[Don't care] check box	Does not set address conditions.
[Address] radio button	Sets the use of the normal address bus as break conditions.
[Only program fetched address after] radio button	Sets a break after prefetched address execution as break conditions.
[Address] edit box	Sets the address bus value with a number or a symbol.
[Mask] edit box	Sets a value to be masked when [User mask] is selected. For masked bits, the break condition is satisfied for any values.

The contents of an option that can be will change depending on the radio button selected.

Option	Description
[Address] radio button	All options can be set.
[Only program fetched address after] radio button	The [Address] option can be set.

The [Data] group box sets the data bus conditions.

Option	Description
[Don't care] check box	No data conditions are set.
[Value] edit box	Sets the data bus with a number.
[Low byte] radio button	Sets data access size as the lower 8-bit access.
[High byte] radio button	Sets data access size as the upper 8-bit access.
[Word] radio button	Sets data access size as the word access.

The [Read/Write] group box sets the read/write cycle conditions.

Option	Description
[Read/Write] radio button	Sets the read/write cycle conditions as break conditions.
[Read] radio button	Sets read cycles as break conditions.
[Write] radio button	Sets write cycles as break conditions.

5.5 Viewing the Trace Information

For the description on the trace function, refer to section 2.2, Trace Functions.

5.5.1 Opening the [Trace] Window

To open the [Trace] window, choose [View -> Code -> Trace] or click the [Trace] toolbar button ( ).

5.5.2 Acquiring Trace Information

The acquired trace information is displayed in the [Trace] window.

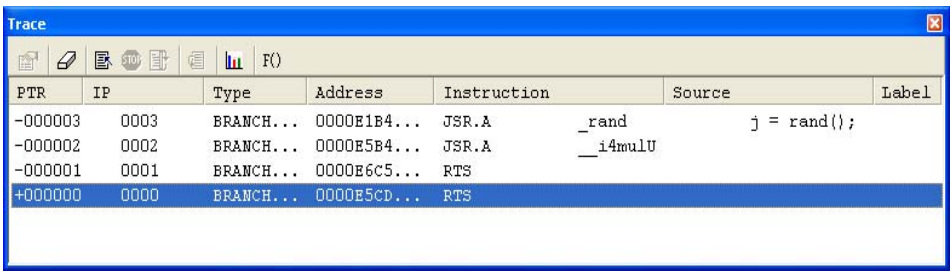

Trace
F()
PTR IP Type Address Instruction Source Label
-000003 0003 BRANCH... 0000E1B4... JSR.A _rand j = rand();
-000002 0002 BRANCH... 0000E5B4... JSR.A __i4mulU
-000001 0001 BRANCH... 0000E6C5... RTS
+000000 0000 BRANCH... 0000E5CD... RTS

Figure 5.9 [Trace] Window

This window displays the following trace information items:

[PTR]	Pointer to a location in the trace buffer (+0 for the last executed instruction)
[IP]	The amount of acquired trace information
[Type]	Type of branch: BRANCH: branch source
[Address]	Instruction address
[Instruction]	Instruction mnemonic
[Source]	The C/C++ or assembly-language source program
[Label]	Label information

It is possible to hide any column not necessary in the [Trace] window. Selecting a column you want to hide from the popup menu displayed by clicking the right-hand mouse button on the header column hides that column. To display the hidden column, select the column from the said popup menu again.

Note: The number of branch instructions that can be acquired by a trace and the trace display differ depending on the product. For the specification of each product, refer to the online help.

5.5.3 Clearing the Trace Information

When [Clear] is selected from the popup menu, the trace buffer that stores the trace information becomes empty. If several [Trace] windows are open, all [Trace] windows will be cleared as they all access the same buffer.

5.5.4 Saving the Trace Information in a File

Select [Save...] from the popup menu to open the [Save As] file dialog box, which allows the user to save the information displayed in the [Trace] window as a text file. A range can be specified based on the [PTR] number. This file can only be saved; it cannot be reloaded into the [Trace] window.

5.5.5 Viewing the [Source] Window

The [Source] window corresponding to the selected trace record can be displayed in the following two ways:

- Select a trace record and choose [View Source] from the popup menu.
- Double-click a trace record

The [Source] or [Disassembly] window opens and the selected line is marked with a cursor.

5.5.6 Trimming the Source

Choose [Trim Source] from the popup menu to remove the white space from the left side of the source.

When the white space is removed, a check mark is shown to the left of the [Trim Source] menu. To restore the white space, choose [Trim Source] while the check mark is shown.

5.6 Using the Start/Stop Function

The emulator executes the specified routine of the user program immediately before starting and immediately after halting program execution. This function is used to control the user system in synchronization with execution and halting of the user program.

Note: The start/stop function is not supported in the H8/Tiny or H8/Super Low Power series, the 740 family, or the R8C/10, R8C/11, R8C/12, or R8C/13 group.

5.6.1 Opening the [Start/Stop Function Setting] Dialog Box

The routine executed immediately before starting and immediately after halting the user program execution is specified in the [Start/Stop Function Setting] dialog box.

To open the [Start/Stop Function Setting] dialog box, choose [Setup -> Emulator -> Start/Stop Function Setting…] from the [Start] menu.

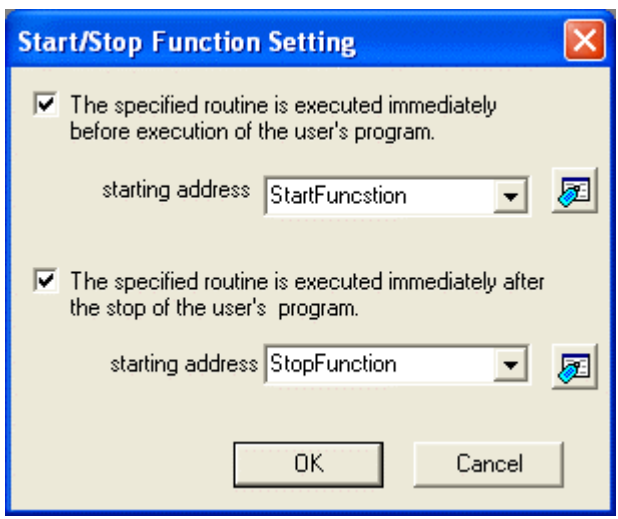

Start/Stop Function Setting
The specified routine is executed immediately before execution of the user's program.
starting address
StartFuncstion
The specified routine is executed immediately after the stop of the user's program.
starting address
StopFunction
OK
Cancel

Figure 5.10 [Start/Stop Function Setting] Dialog Box

5.6.2 Specifying the Routine to be Executed

It is possible to specify the respective routines immediately before starting and immediately after halting the user program execution.

When the [The specified routine is executed immediately before execution of the user's program.] check box is selected, the routine specified in the [starting address] combo box, which is below this check box, is executed immediately before starting user program execution.

If no specified routine is executed immediately before starting user program execution, do not select the [The specified routine is executed immediately before execution of the user's program.] check box.

When the [The specified routine is executed immediately after the stop of the user's program.] check box is selected, the routine specified in the [starting address] combo box, which is below this check box, is executed immediately after halting user program execution.

If no specified routine is executed immediately after halting user program execution, do not select the [The specified routine is executed immediately after the stop of the user's program.] check box.

5.6.3 Restrictions on the Start/Stop Function

The start/stop function has the following restrictions.

- Do not change the SFR used by the firmware.
- When the watchdog timer is used, refresh the watchdog timer in the specified routine.
- Do not set any breakpoints in the specified routine.
- When a stack is used in the specified routine, be sure to use the user stack.
- A register value is undefined when execution of the specified routine is started. It is possible to change the register value in the specified routine, however, the following register and flags have restrictions.

Register and Flag Names	Restrictions
ISP register	When execution of the specified routine is ended, put the value back to the one at the start of that routine.
Flag U	Be sure to set the flag value as 0 when execution of the specified routine is ended.
Flag B	Do not set the flag value as 1 in the specified routine.
Flag I	No interrupts are allowed during execution of the specified routine. Do not set the flag value as 1 in the specified routine.

- When the content of the [Start/Stop Function Setting] dialog box is changed, the flash memory will be reprogrammed in the block which the firmware has been allocated at user program execution.
- The specified routine is also executed when the following operations are performed:
  — Step operations
  — Watching or changing memory during user program execution
  — Setting an event break during user program execution

Section 6 Tutorial

6.1 Introduction

This section describes the main functions of the emulator by using a tutorial program.

The tutorial program is based on the C program that sorts ten random data items in ascending or descending order. The tutorial program performs the following actions:

- The `main` function generates random data to be sorted.
- The `sort` function sorts the generated random data in ascending order.
- The `change` function then sorts the data in descending order.

The file `tutorial.c` contains source code for the tutorial program. The file `Tutorial.x30` is a compiled load module in the IEEE695 format.

Notes:

1. After recompilation, the addresses may differ from those given in this section.
2. The file `Tutorial.cpp` contains source code for the tutorial program of the H8 family devices. The file `Tutorial.abs` is a compiled load module in the Elf/Dwarf2 format.

6.2 Running the High-performance Embedded Workshop

To run the High-performance Embedded Workshop, refer to section 3.8, System Check.

6.3 Setting up the E8a Emulator

This section describes the basic settings of the emulator.

6.4 Setting the [Configuration] Dialog Box

- Select [Emulator] then [Systems…] from the [Setup] menu for the basic settings of the emulator. The [Configuration] dialog box is displayed.

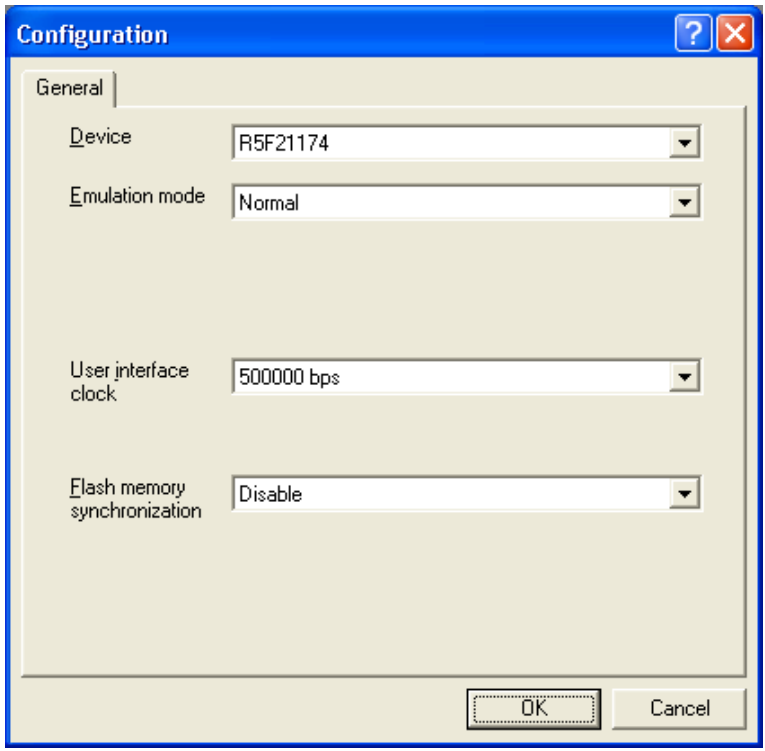


Figure 6.1 [Configuration] Dialog Box

Note: The items that can be set in this dialog box differ depending on the product. For the settings for each product, refer to the online help.

Set options as follows:

Table 6.1 [General] Page Options

Option	Value
Device	Select a device you use
Emulation mode	Normal (normal execution, default)
User interface clock	According to the system clock frequency, the maximum user interface clock is automatically set.
Flash memory synchronization	Disable (default)

Click the [OK] button to set a configuration.

6.5 Checking the Operation of RAM

Check that RAM is operating correctly.

Display and edit the contents of the memory in the [Memory] window to check that the memory is operating correctly.

Note: The memory can be installed on the board in some microcomputers. In this case, however, the above way of checking the operation of memory may be inadequate. It is recommended that a program for checking the memory be created.

- Select [Memory…] from the [CPU] submenu of the [View] menu and enter the RAM address in the [Display Address] edit box. Here, enter ***'0480'***. Then, enter ***'0000'*** and ***'FFFF'*** in the [Scroll Start Address] and [Scroll End Address] edit boxes, respectively.

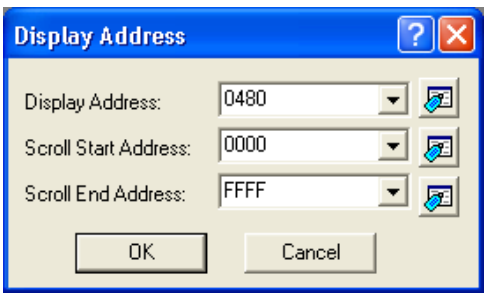


Figure 6.2 [Display Address] Dialog Box

Note: The settings of the RAM area differ depending on the product. For details, refer to the hardware manual.

- Click the [OK] button. The [Memory] window is displayed and shows the specified memory area.

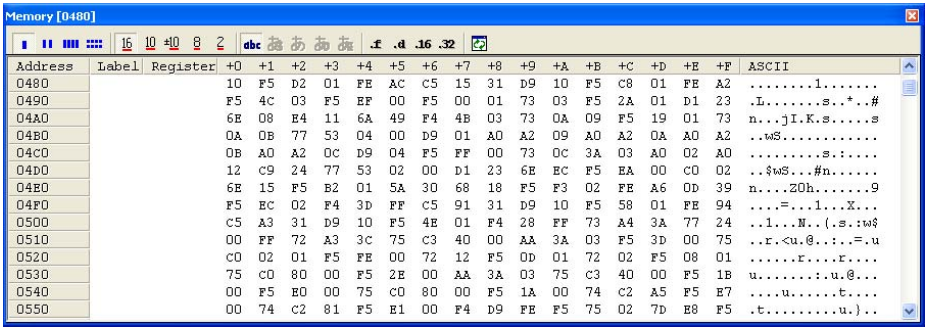


Figure 6.3 [Memory] Window

- Placing the mouse cursor on a point in the display of data in the [Memory] window and double-clicking allows the values at that point to be changed. Data can also be directly edited around the current position of the text cursor.

6.6 Downloading the Tutorial Program

6.6.1 Downloading the Tutorial Program

Download the object program to be debugged.

In this emulator, it is enabled to download the program and set the PC breakpoint in the internal flash memory area. For the method to set the PC breakpoint, refer to section 6.16.1, PC Break Function.

- Select [Download] from [Tutorial.x30] under [Download modules].

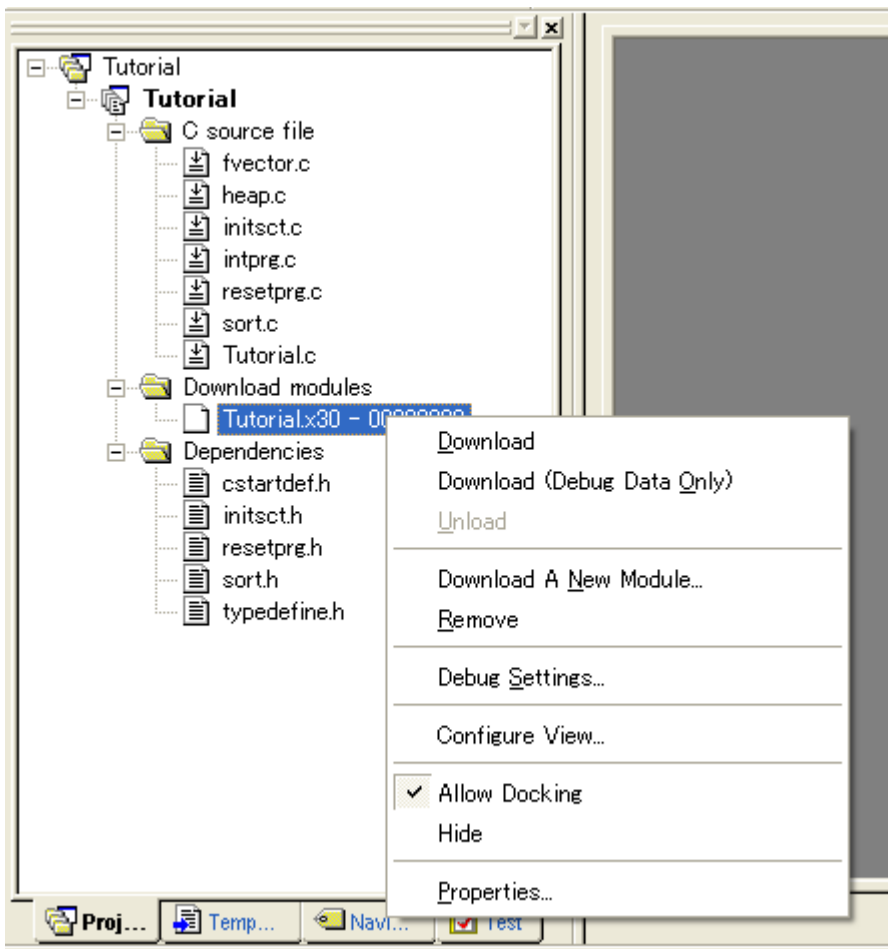


Figure 6.4 Downloading the Tutorial Program

6.6.2 Displaying the Source Program

The High-performance Embedded Workshop allows the user to debug a user program at the source level.

- Double-click [tutorial.c] under [C source file].

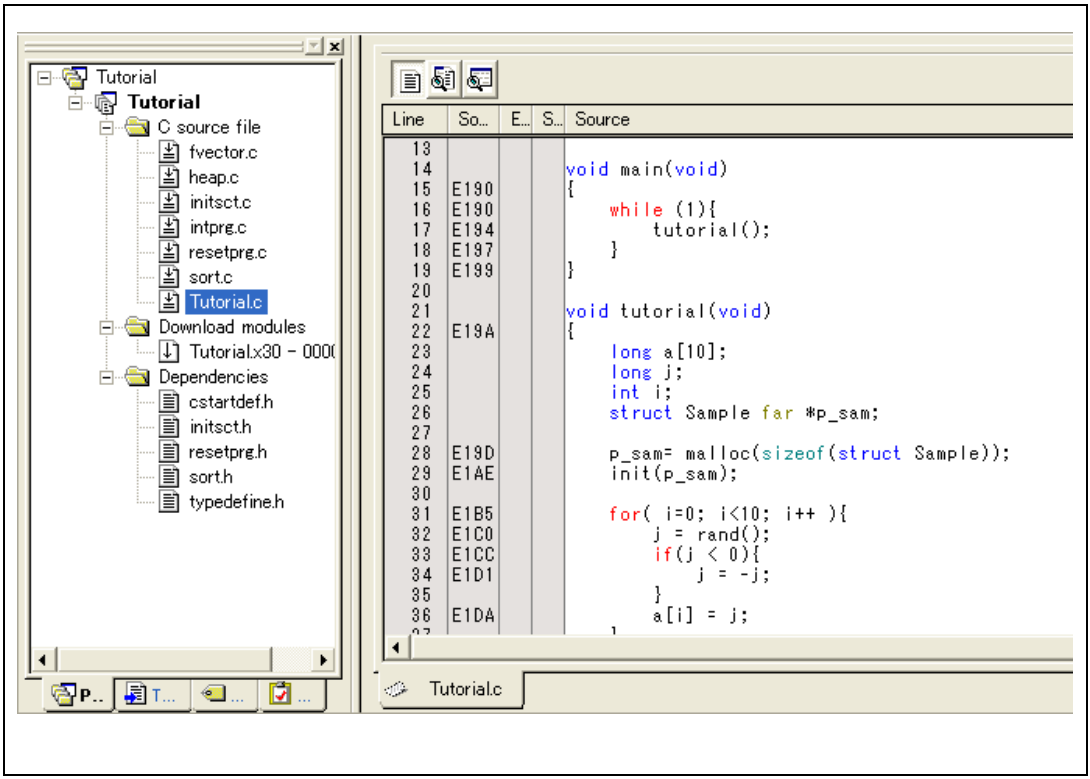


Figure 6.5 [Source] Window (Displaying the Source Program)

Select the [Display] option from the [Setup] menu to set a font and size that are legible, if necessary.
Initially the [Source] window shows the start of the user program, but the user can use the scroll bar to scroll through the user program and look at the other statements.

6.7 Setting a PC Breakpoint

A PC breakpoint is a simple debugging function.

The [Source] window provides a very simple way of setting a PC breakpoint at any point in a program. For example, to set a PC breakpoint at the `sort` function call:

- Select by double-clicking the [S/W Breakpoints] column on the line containing the `sort` function call.

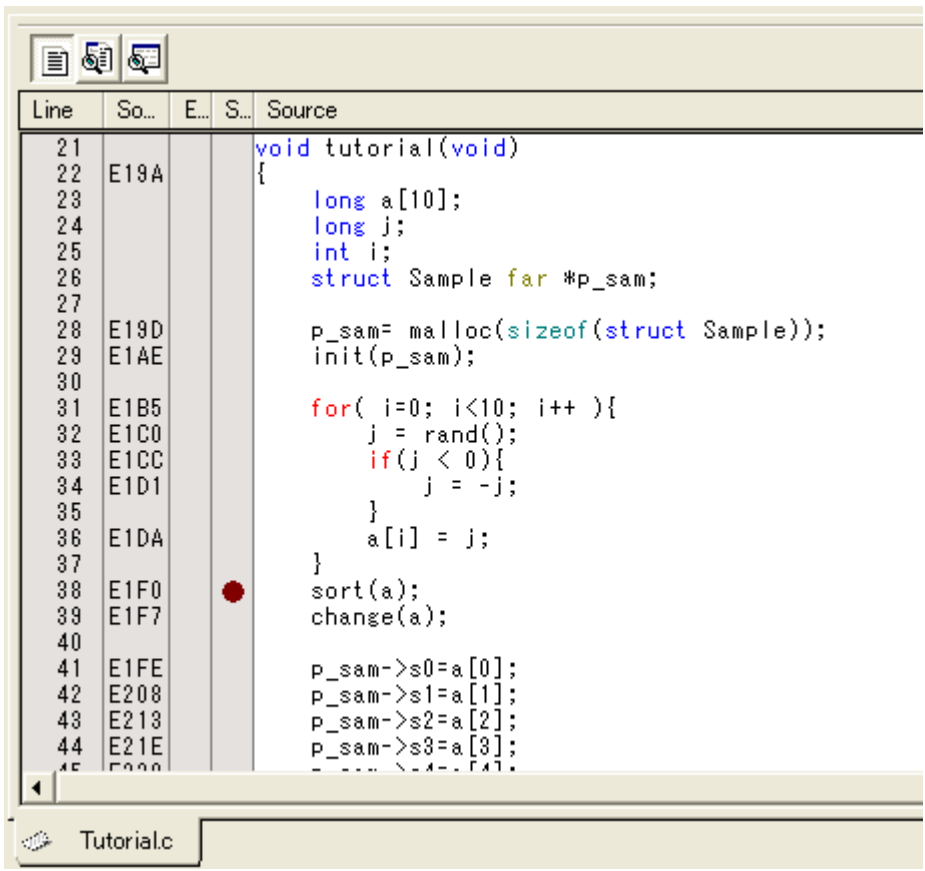


Figure 6.6 [Source] Window (Setting a PC Breakpoint)

The symbol • will appear on the line containing the `sort` function. This shows that a PC breakpoint has been set.

6.8 Setting Registers

Set values of the program counter and the stack pointer before executing the program.

- Select [Registers] from the [CPU] submenu of the [View] menu. The [Register] window is displayed.

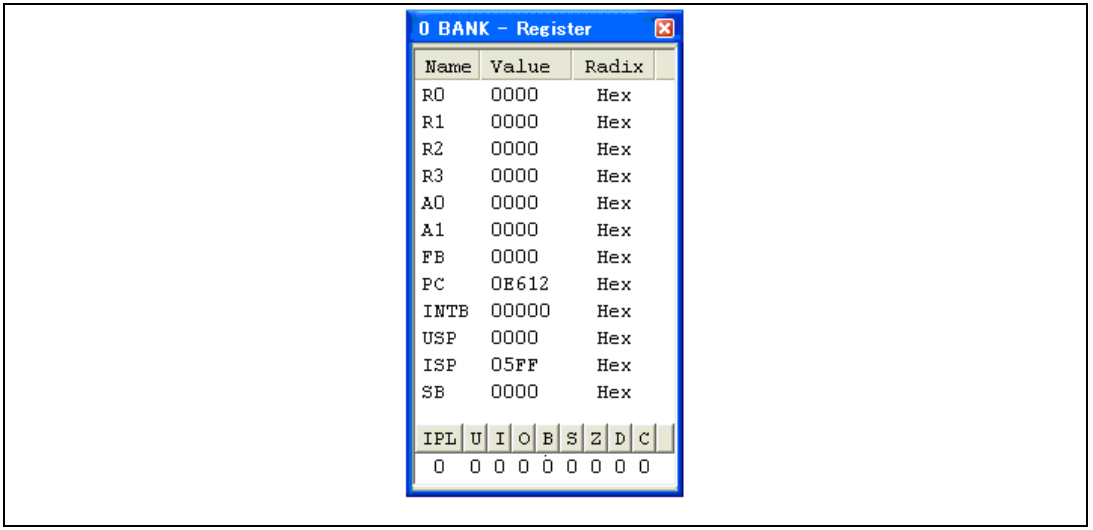


Figure 6.7 [Register] Window

- To change the value of the program counter (PC), double-click the value area in the [Register] window with the mouse. The following dialog box is then displayed, and the value can be changed. Set the program counter to 0E612 in this tutorial program, and click the [OK] button.

Figure 6.8 [Register] Dialog Box (PC)

6.9 Executing the Program

Execute the program as described in the following:

- To execute the program, select [Go] from the [Debug] menu, or click the [Go] button on the toolbar.

Figure 6.9 [Go] Button

When the program execution is started, `'**RUNNING'` is displayed on the status bar.

The program will be executed up to the breakpoint that has been set, and an arrow will be displayed in the [S/W Breakpoints] column to show the position that the program has halted, with the message `[BREAKPOINT]` in the status bar.

Note: When the source file is displayed after a break, a path of the source file may be inquired. The location of the source file is as follows:

The High-performance Embedded Workshop is installed to:
\WorkSpace\Tutorial\E8a\xxxx\Tutorial\Tutorial\Source

The directory xxxx varies depending on each device.
e.g.: M32C
\WorksSpace\Tutorial\E8a\M32C\Tutorial\Tutorial\Source

e.g.: M16C
\WorksSpace\Tutorial\E8a\M16C\Tutorial\Tutorial\Source

e.g.: R8C/Tiny
\WorksSpace\Tutorial\E8a\R8C\Tutorial\Tutorial\Source

e.g.: 740
\WorksSpace\Tutorial\E8a\740\Tutorial\Tutorial\Source

e.g.: H8/Tiny normal mode
\WorksSpace\Tutorial\E8a\H8\Tutorial

e.g.: H8/Tiny advanced mode
\WorksSpace\Tutorial\E8a\H8\TutorialADV

e.g.: H8/Super Low Power
\WorksSpace\Tutorial\E8a\H8\TutorialSLP

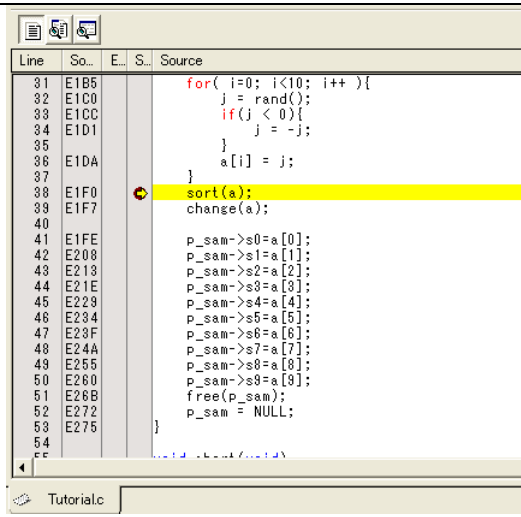


Figure 6.10 [Source] Window (Break State)

The user can see the cause of the break that occurred last time in the [Status] window.

- Select [Status] from the [CPU] submenu of the [View] menu. After the [Status] window is displayed, open the [Platform] sheet, and check the `Status` of `Cause of last break`.

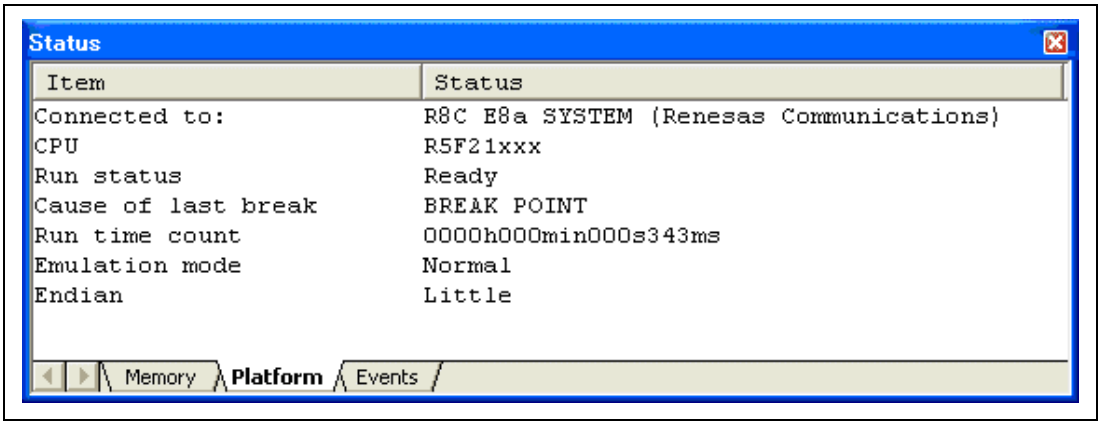


Figure 6.11 [Status] Window

Note: The items that can be set in this window differ depending on the product. For the settings for each product, refer to the online help.

6.10 Reviewing Breakpoints

The user can see all the breakpoints set in the program in the [Event] window.

- Select [Eventpoints] from the [Code] submenu of the [View] menu. The [Event] window is displayed. Select the [Breakpoint] sheet.

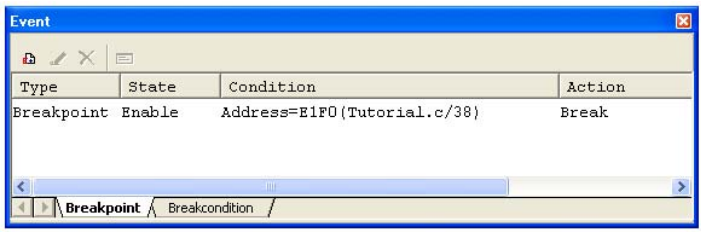


Figure 6.12 [Event] Window

The popup menu, opened by clicking the [Event] window with the right-hand mouse button, allows the user to set or change breakpoints, define new breakpoints, and delete, enable, or disable breakpoints.

6.11 Viewing Memory

When the label name is specified, the user can view the memory contents that the label has been registered in the [Memory] window. For example, to view the memory contents corresponding to `_main` in word size:

- Select [Memory …] from the [CPU] submenu of the [View] menu, enter ***_main*** in the [Display Address] edit box, enter ***0000*** and ***FFFF*** in the [Scroll Start Address] and [Scroll End Address] edit boxes, respectively.

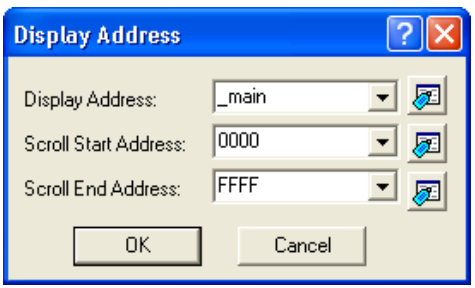


Figure 6.13 [Display Address] Dialog Box

- Click the [OK] button. The [Memory] window showing the specified area of memory is displayed.

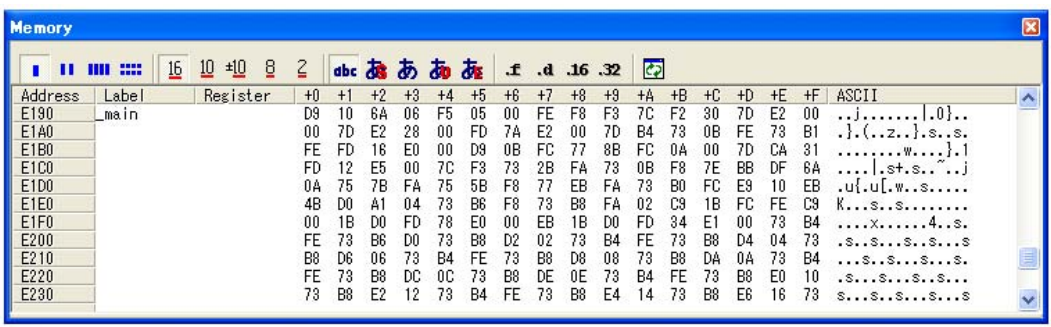


Figure 6.14 [Memory] Window

6.12 Watching Variables

As the user steps through a program, it is possible to watch that the values of variables used in the user program are changed. For example, set a watch on the long-type array `a` declared at the beginning of the program, by using the following procedure:

- Click the left of displayed array `a` in the [Source] window to position the cursor.
- Select [Instant Watch...] with the right-hand mouse button.

The following dialog box will be displayed.

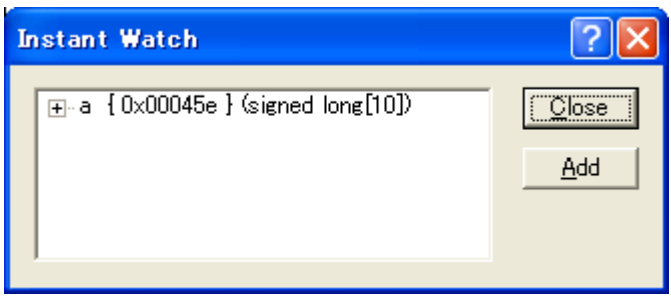


Figure 6.15 [Instant Watch] Dialog Box

- Click the [Add] button to add a variable to the [Watch] window.

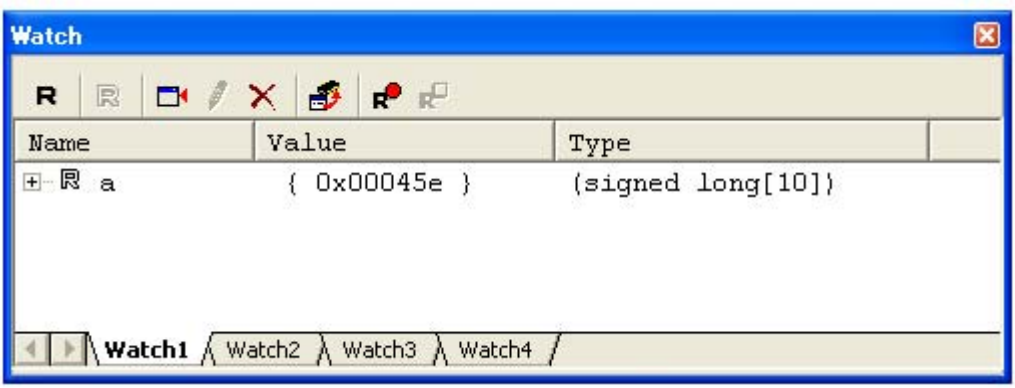


Figure 6.16 [Watch] Window (Displaying the Array)

The user can also add a variable to the [Watch] window by specifying its name.

- Click the [Watch] window with the right-hand mouse button and select [Add Watch…] from the popup menu.

The following dialog box will be displayed. Enter variable `i`.

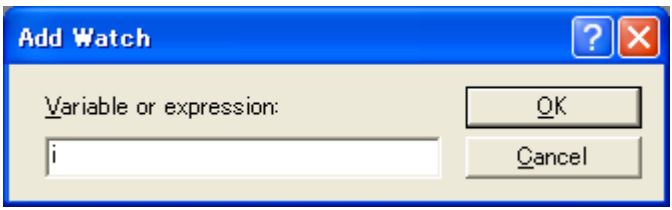


Figure 6.17 [Add Watch] Dialog Box

- Click the [OK] button.

The [Watch] window will now also show the int-type variable `i`.

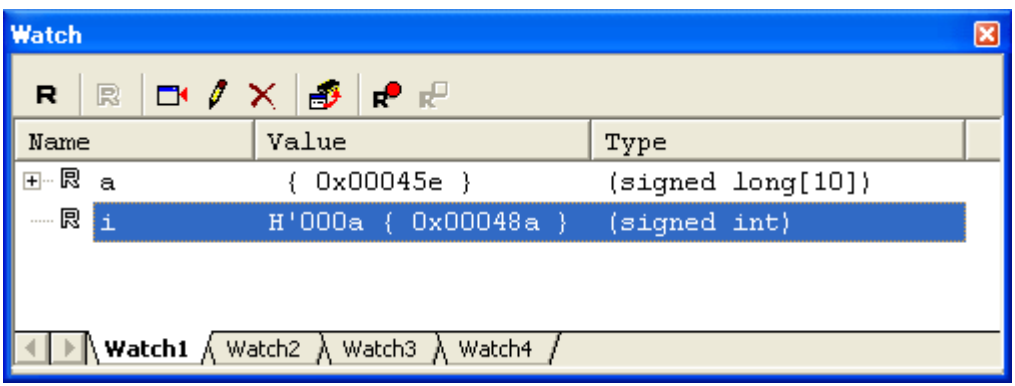


Figure 6.18 [Watch] Window (Displaying the Variable)

The user can click mark '+' at the left side of array a in the [Watch] window to watch all the elements.

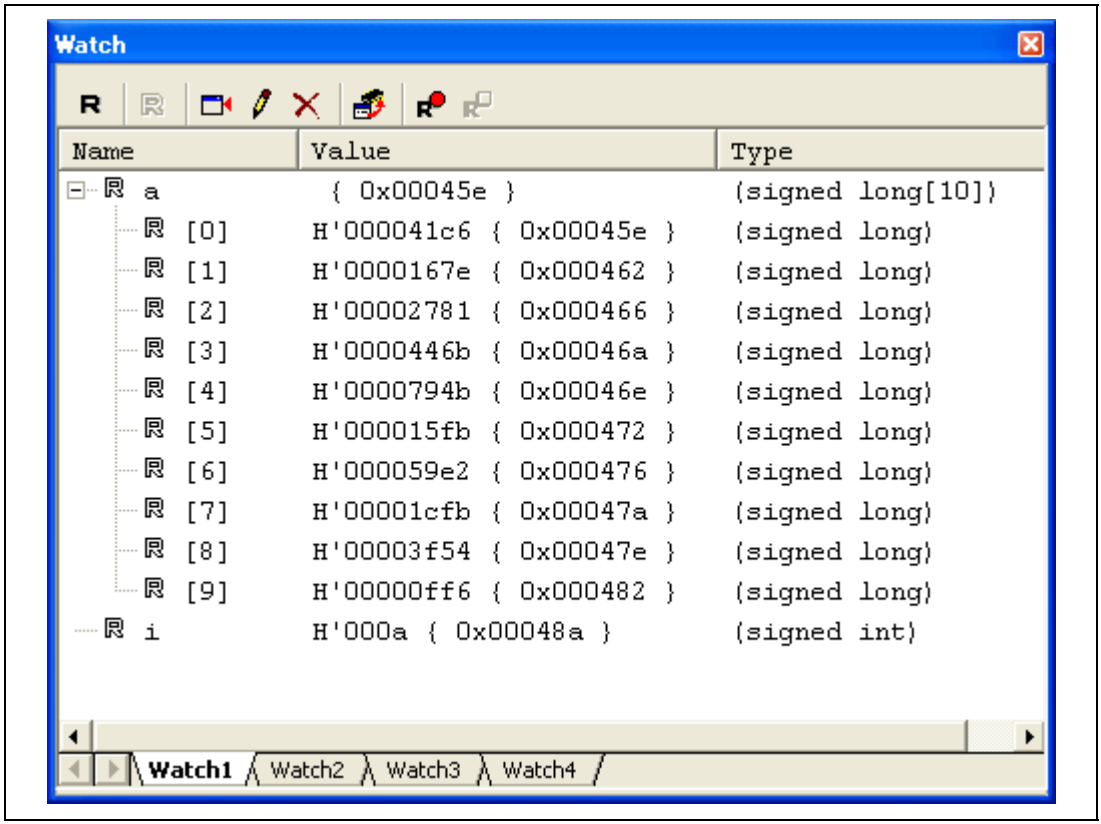

Watch
Name
Value
Type
a { 0x00045e } (signed long[10])
[0] H'000041c6 { 0x00045e } (signed long)
[1] H'0000167e { 0x000462 } (signed long)
[2] H'00002781 { 0x000466 } (signed long)
[3] H'0000446b { 0x00046a } (signed long)
[4] H'0000794b { 0x00046e } (signed long)
[5] H'000015fb { 0x000472 } (signed long)
[6] H'000059e2 { 0x000476 } (signed long)
[7] H'00001cfb { 0x00047a } (signed long)
[8] H'00003f54 { 0x00047e } (signed long)
[9] H'00000ff6 { 0x000482 } (signed long)
i H'000a { 0x00048a } (signed int)
Watch1
Watch2
Watch3
Watch4

Figure 6.19 [Watch] Window (Displaying Array Elements)

6.13 Stepping Through a Program

The High-performance Embedded Workshop provides a range of step menu commands that allow efficient program debugging.

Table 6.2 Step Option

Menu Command	Description
Step In	Executes each statement, including statements within functions.
Step Over	Executes a function call in a single step.
Step Out	Steps out of a function, and stops at the statement following the statement in the program that called the function.
Step…	Steps the specified times repeatedly at a specified rate.

6.13.1 Executing [Step In] Command

The [Step In] command steps into the called function and stops at the first statement of the called function.

- To step through the `sort` function, select [Step In] from the [Debug] menu, or click the [Step In] button on the toolbar.

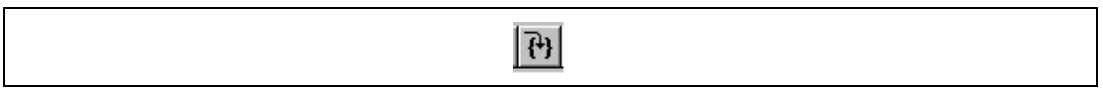


Figure 6.20 [Step In] Button

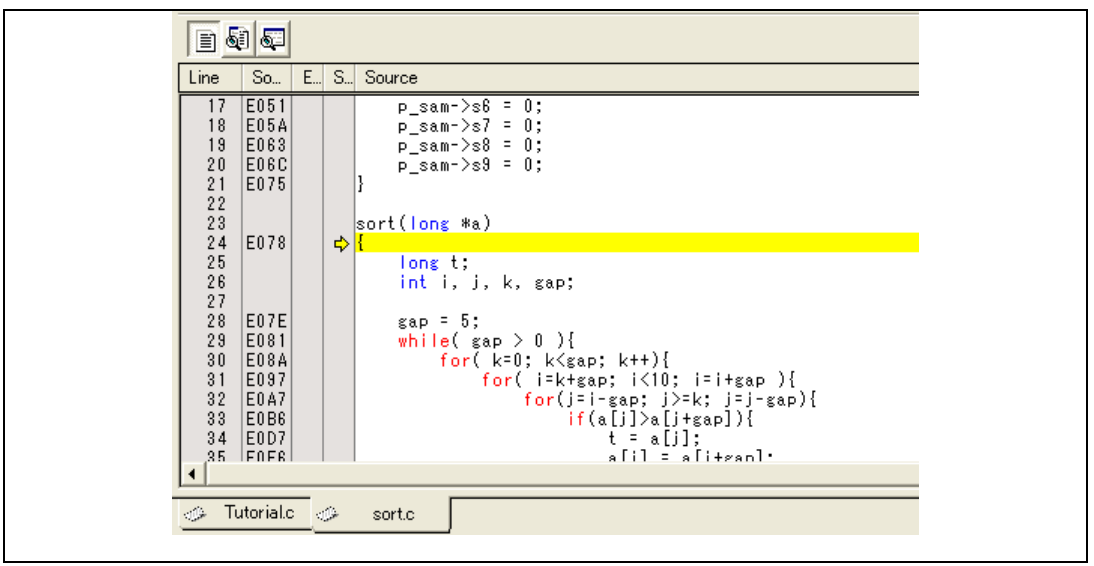


Figure 6.21 [Source] Window (Step In)

- The highlighted line moves to the first statement of the `sort` function in the [Source] window.

6.13.2 Executing [Step Out] Command

The [Step Out] command steps out of the called function and stops at the next statement of the calling statement in the main function.

- To step out of the `sort` function, select [Step Out] from the [Debug] menu, or click the [Step Out] button on the toolbar.

Note: It takes time to execute this function. When the calling source is clarified, use [Go To Cursor].

Figure 6.22 [Step Out] Button

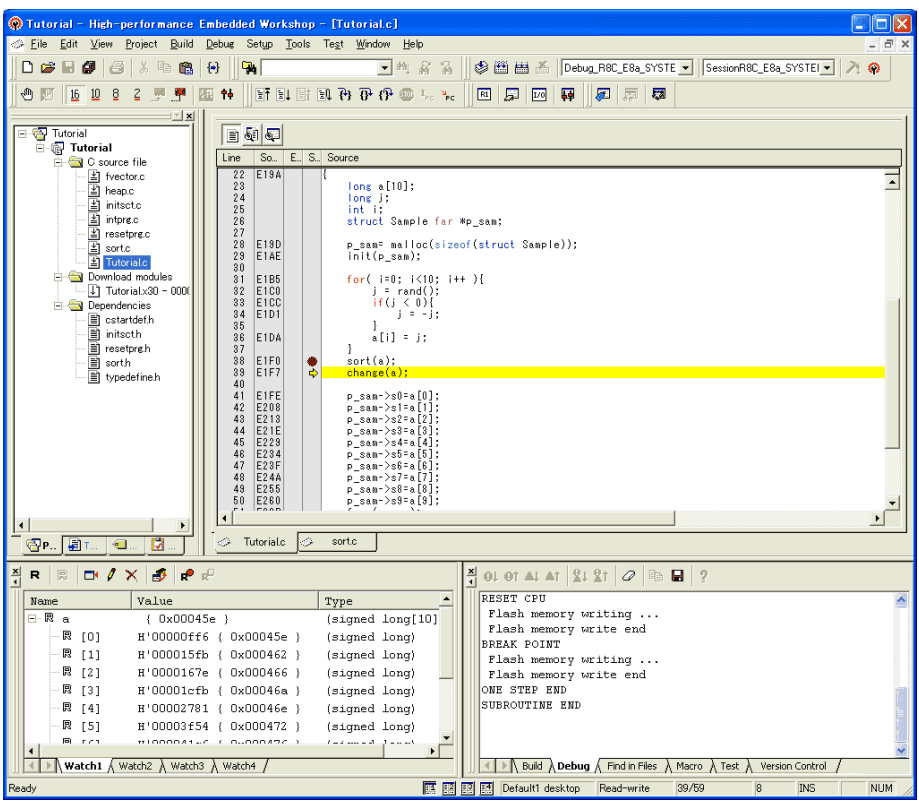


Figure 6.23 [High-performance Embedded Workshop] Window (Step Out)

The data of variable `a` displayed in the [Watch] window is sorted in ascending order.

6.13.3 Executing [Step Over] Command

The [Step Over] command executes a function call as a single step and stops at the next statement of the main program.

- To step through all statements in the `change` function at a single step, select [Step Over] from the [Debug] menu, or click the [Step Over] button on the toolbar.

Figure 6.24 [Step Over] Button

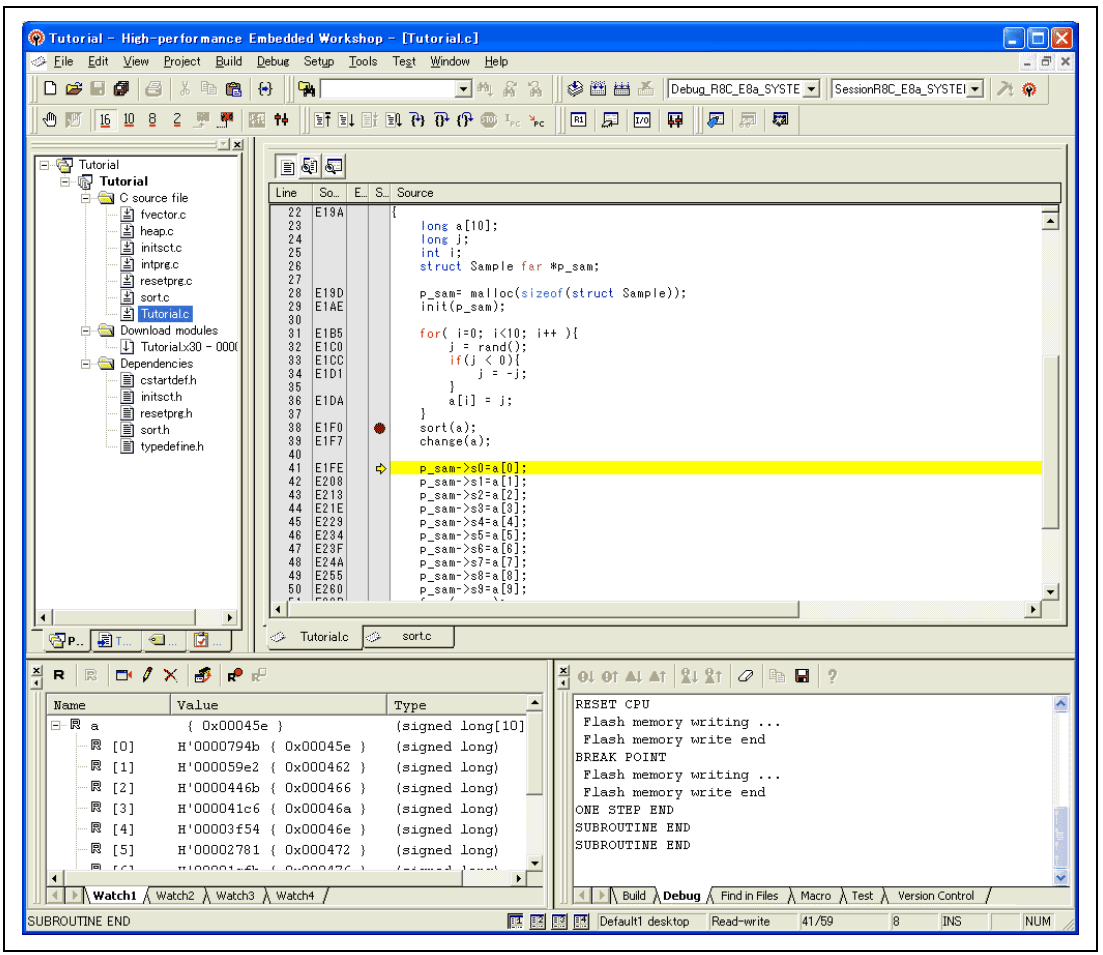


Figure 6.25 [High-performance Embedded Workshop] Window (Step Over)

6.14 Forced Breaking of Program Executions

The High-performance Embedded Workshop can force a break in the execution of a program.

- Cancel all breaks.
- To execute the remaining sections of the `main` function, select [Go] from the [Debug] menu or the [Go] button on the toolbar.

Figure 6.26 [Go] Button

- The program goes into an endless loop. To force a break in execution, select [Halt] from the [Debug] menu or the [Halt] button on the toolbar.

Figure 6.27 [Halt] Button

6.15 Displaying Local Variables

The user can display local variables in a function using the [Locals] window. For example, we will examine the local variables in the `main` function, which declares four local variables: `a`, `j`, `i`, and `p_sam`.

- Select [Locals] from the [Symbol] submenu of the [View] menu. The [Locals] window is displayed.

The [Locals] window shows the local variables in the function currently pointed to by the program counter, along with their values. Note, however, that the [Locals] window is initially empty because local variables are yet to be declared.

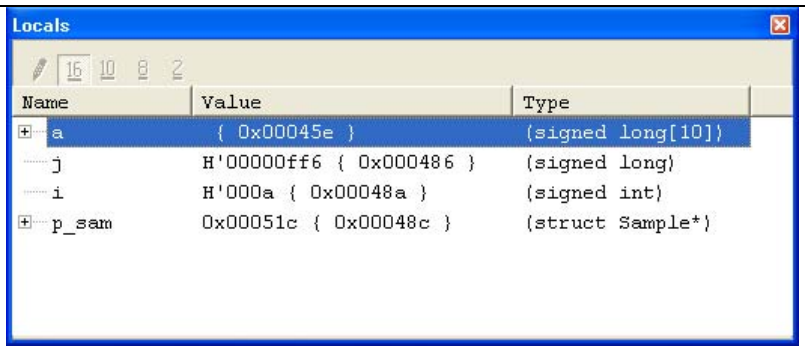


Figure 6.28 [Locals] Window

- Click mark '+' at the left side of array `a` in the [Locals] window to display the elements.
- When the user refers to the elements of array `a` before and after the execution of the `sort` function, it is clarified that random data is sorted in descending order.

6.16 Break Function

The emulator has PC and hardware break functions. With the High-performance Embedded Workshop, a PC breakpoint can be set using the [Breakpoint] sheet of the [Event] window, and a hardware break condition can be set using the [Breakcondition] sheet.

An overview and setting of the break function are described below.

6.16.1 PC Break Function

The emulator can set up to 255 PC breakpoints. Other methods for setting a PC breakpoint than in section 6.7, Setting a PC Breakpoint, are described below.

- Select [Eventpoints] from the [Code] submenu of the [View] menu. The [Event] window is displayed.
- Select the [Breakpoint] sheet.

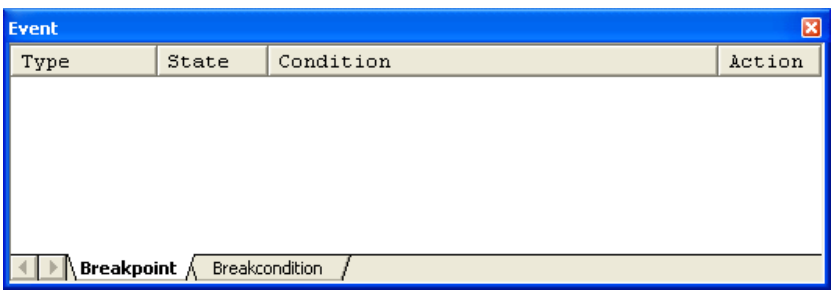


Figure 6.29 [Event] Window (Before PC Breakpoint Setting)

- Click the [Event] window with the right-hand mouse button and select [Add…] from the popup menu.
- Enter ***H'0E1FE*** in the [Address] edit box.

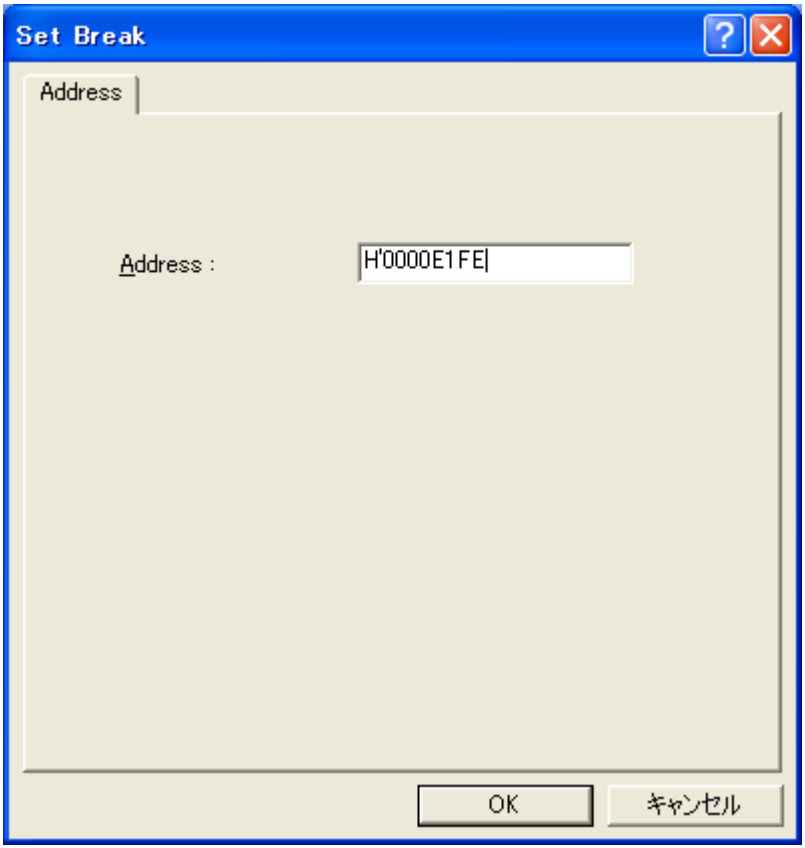


Figure 6.30 [Set Break] Dialog Box

- Click the [OK] button.

The PC breakpoint that has been set is displayed in the [Event] window.

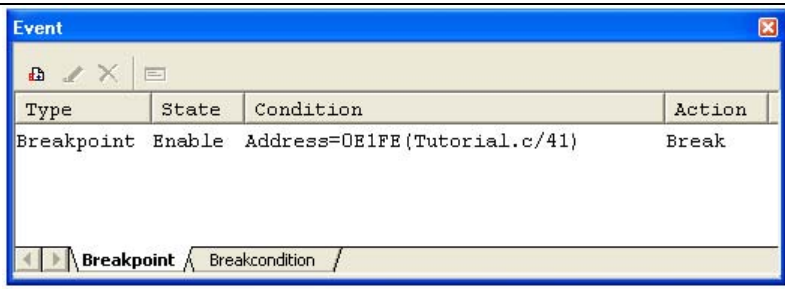


Figure 6.31 [Event] Window (PC Breakpoint Setting)

To stop the tutorial program at the PC breakpoint, the following procedure must be executed:

- Close the [Event] window.
- Set the program counter value (PC = H'0E612) that was set in section 6.8, Setting Registers, in the [Register] window. Click the [Go] button.
- If program execution is failed, reset the device and execute again the procedures above.

The program runs, and stops at the set PC breakpoint.

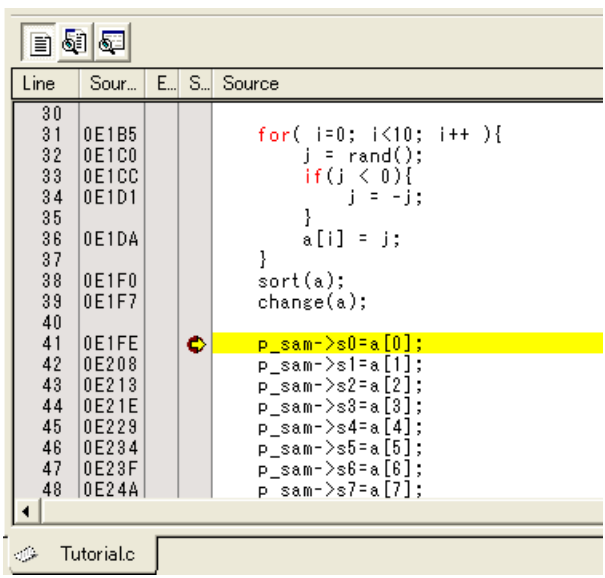


Figure 6.32 [Source] Window at Execution Stop (PC Break)

The [Status] window displays the following contents.

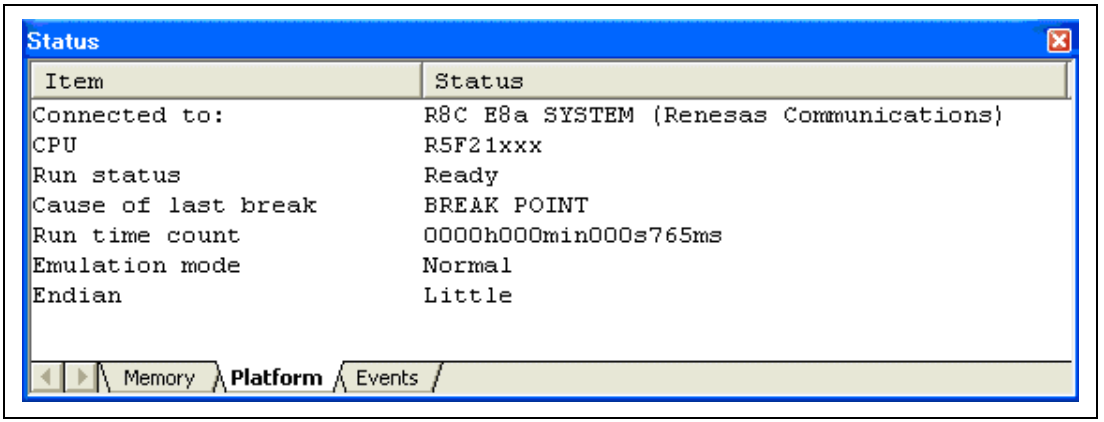


Figure 6.33 Displayed Contents of the [Status] Window (PC Break)

Note: The items that can be set in this window differ depending on the product. For the settings for each product, refer to the online help.

6.17 Hardware Break Function

A method is given below in which the address bus condition is set under Break Condition 1 as hardware break conditions.

- Select [Eventpoints] from the [Code] submenu of the [View] menu. The [Event] window is displayed.
- The PC breakpoint that has been previously set is deleted. Click the [Event] window with the right-hand mouse button and select [Delete All] from the popup menu to cancel all PC breakpoints that have been set.
- To set a Break condition 1, click the [Break condition] tab.
- Select a line of Break condition 1 in the [Event] window. When highlighted, double-click this line.

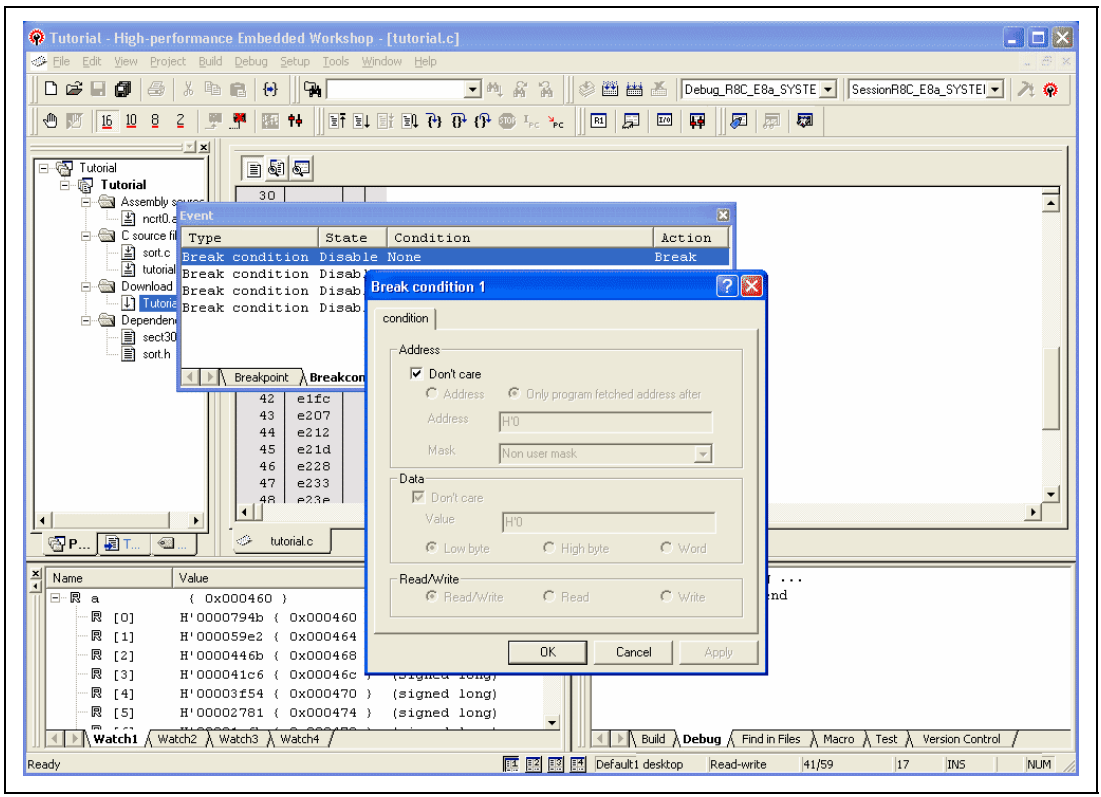


Figure 6.34 [High-performance Embedded Workshop] Window ([Break condition 1])

- The [Break condition 1] dialog box is displayed.
- Clear the [Don't care] check box in the [Address] group box.
- Select the [Only program fetched address after] radio button and enter ***H'0E1F2*** as the value in the [Address] edit box.

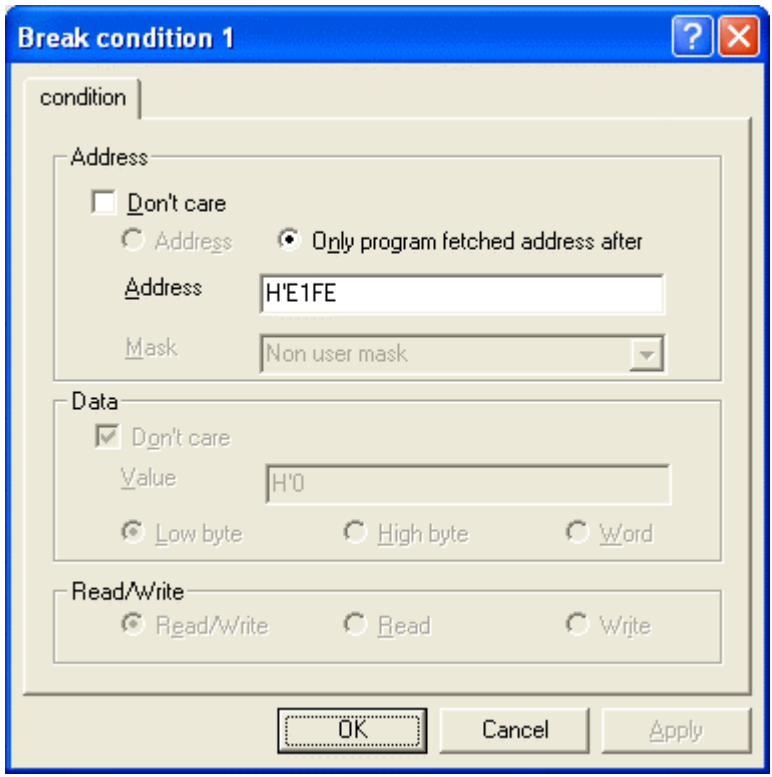


Figure 6.35 [condition] Page ([Break condition 1] Dialog Box)

Click the [OK] button.

The first point display in the State line changes from `Disable` to `Enable`.

The first point display in the Condition line changes from `None` to `Address = H'0000E1FE(Tutorial.c/41)pcbefore`.

Set the program counter value (PC = H'0E612) that was set in section 6.8, Setting Registers, in the [Register] window. Click the [Go] button.

Note: The items that can be set in this page differ depending on the product. For the settings for each product, refer to the online help.

The program runs and then stops at the condition specified under Break Condition 1.

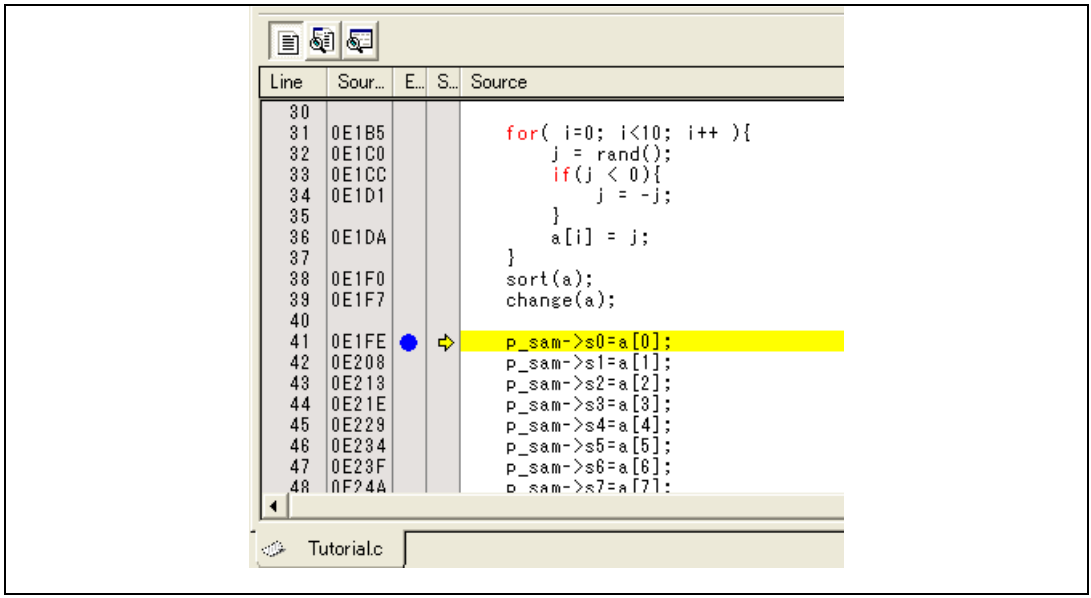


Figure 6.36 [Source] Window at Execution Stop (Break Condition 1)

The [Status] window displays the following contents.

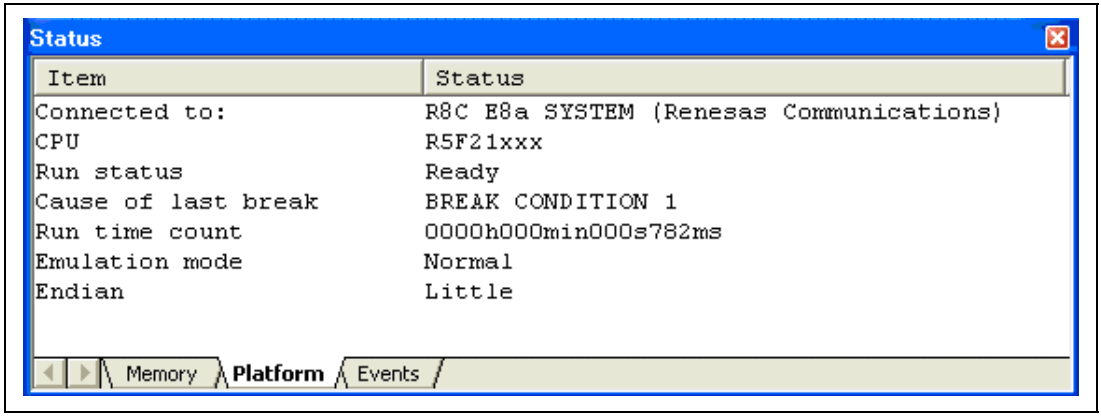


Figure 6.37 Displayed Contents of the [Status] Window (Break Condition 1)

Note: The items that can be set in this window differ depending on the product. For the settings for each product, refer to the online help.

6.18 Trace Functions

The emulator has a branch-instruction trace function.

The branch source addresses, mnemonics, operands, source lines, and labels are displayed for four branches.

Note: The trace function uses the trace buffer housed in the device.
Devices which have no trace function cannot use this function.

6.18.1 Displaying the Trace Window

Select [Trace] from the [Code] submenu of the [View] menu.

Run the program as shown in the example of section 6.16.1, PC Break Function. The trace results are displayed in the [Trace] window after the program execution is completed.

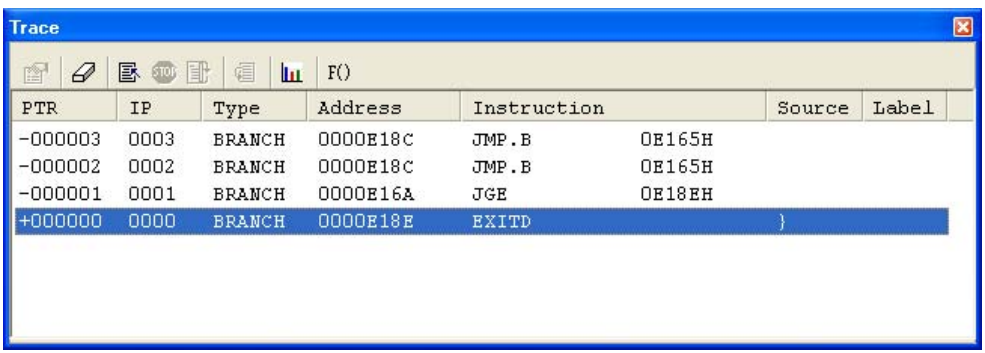

Trace
F()
PTR IP Type Address Instruction Source Label
-000003 0003 BRANCH 0000E18C JMP.B 0E165H
-000002 0002 BRANCH 0000E18C JMP.B 0E165H
-000001 0001 BRANCH 0000E16A JGE 0E18EH
+000000 0000 BRANCH 0000E18E EXITD }

Figure 6.38 [Trace] Window

If necessary, adjust the column widths by dragging borders in the header bar (immediately below the title bar).

Note: The number of branch instructions that can be acquired by a trace and the trace display differ depending on the product. For the specification of each product, refer to the online help.

6.19 Stack Trace Function

The emulator uses the information on the stack to display the names of functions in the sequence of calls that led to the function to which the program counter is currently pointing.

Note: This function can be used only when the load module that has the IEEE695-type debugging information is loaded.（However, 740 families exclude it.）

- Double-click the [S/W Breakpoints] column in the `sort` function and set a PC breakpoint.

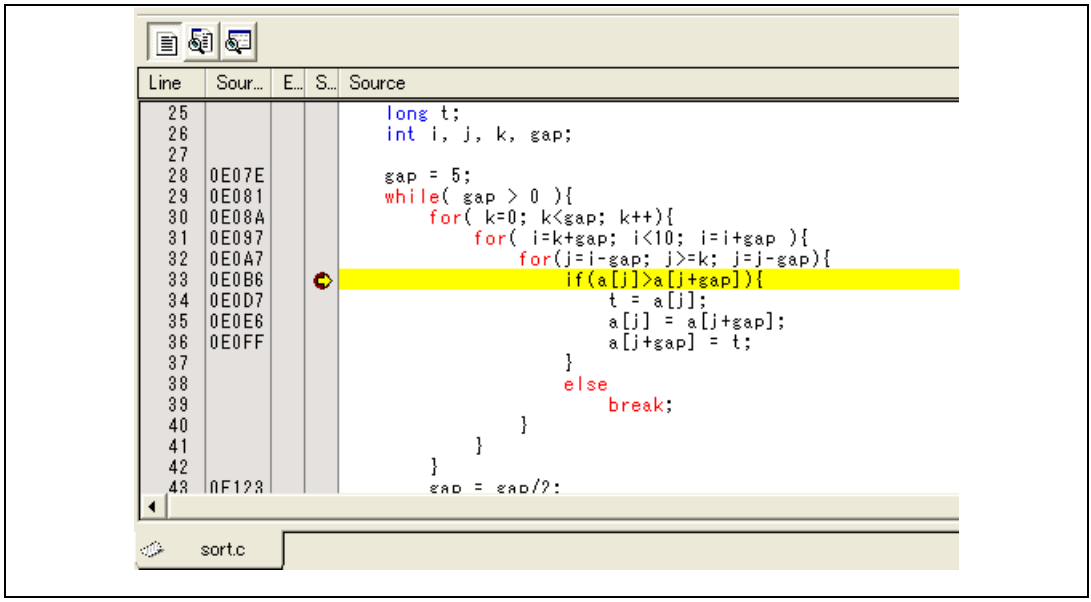


Figure 6.39 [Source] Window (PC Breakpoint Setting)

- Set the same program counter value (PC = H’0E612) as was set in section 6.8, Setting Registers (again, use the [Register] window). Click the [Go] button.

- After the break in program execution, select [Stack Trace] from the [Code] submenu of the [View] menu to open the [Stack Trace] window.

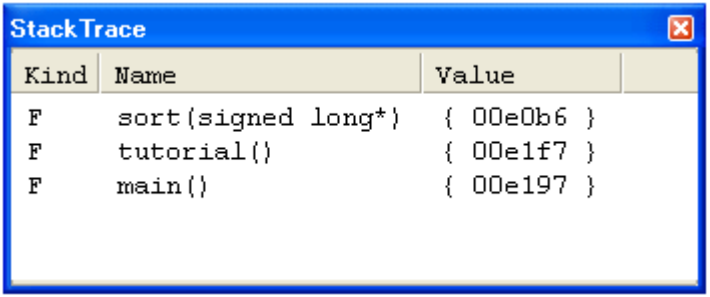


Figure 6.40 [Stack Trace] Window

Figure 6.40 shows that the position of the program counter is currently at the selected line of the `sort()` function, and that the `sort()` function is called from the `tutorial()` function.

6.20 What Next?

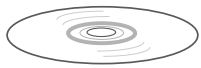
This tutorial has described the major features of the emulator and the use of the High-performance Embedded Workshop.

Sophisticated debugging can be carried out by using the emulation functions that the emulator offers. This provides for effective investigation of hardware and software problems by accurately isolating and identifying the conditions under which such problems arise.

Appendix A Components of the E8a Emulator

Table A.1 lists the components of the E8a emulator.

Table A.1 Components of the Emulator (R0E00008AKCE00)

Classi-fication	Component	Appearance	Quantity	Remarks
Hard-ware	E8a emulator (Model: R0E00008AKCE00)		1	R0E00008AKCE00: Depth: 65 mm, Width: 97 mm, Height: 21 mm, Mass: 60.0 g
	User system interface cable		1	Length: 200 mm (connector part not included), Mass: 20.0 g
	USB cable		1	Length: 1.5 m, Mass: 52.0 g
Soft-ware	E8a emulator setup program,		1	R0E00008AKCE00SR,
	E8a emulator self-check program,			E8aSCP.EXE,
	E8a Emulator User's Manual,			R0E00008AKCE00J, R0E00008AKCE00E,
	E8a Emulator Additional Document for User's Manual, Notes on Connecting the xxxxx			R0E00008AKCE00JPn, R0E00008AKCE00EPn (n = 1, 2, 3, 4, …),
	M32C evaluation compiler M16C evaluation compiler 740 evaluation compiler 740 evaluation assembler and H8 evaluation compiler			M3T-NC308WA M3T-NC30WA M3T-ICC740 M3T-SRA74 and R0C40008XSW06R (Provided with the E8a emulator setup program)

Appendix B Window Functions

This section describes the window functions that can be used with the High-performance Embedded Workshop for E8a emulator. Here, the High-performance Embedded Workshop common functions are omitted, and only the emulation functions are described.

Table B.1 Window Functions

Menu	Option		Shortcut	Toolbar Button	Remarks
View	Disassembly		Ctrl + D		Opens the [Disassembly] window.
	Command Line		Ctrl + L		Opens the [Command Line] window.
	Workspace		Alt + K		Opens the [Workspace] window.
	Output		Alt + U		Opens the [Output] window.
	CPU	Registers	Ctrl + R		Opens the [Register] window.
		Memory…	Ctrl + M		Opens the [Memory] window.
		IO	Ctrl + I		Opens the [IO] window.
		Status	Ctrl + U		Opens the [Status] window.
	Symbol	Labels	Shift + Ctrl + A		Opens the [Labels] window.
		Watch	Ctrl + W		Opens the [Watch] window.
		Locals	Shift + Ctrl + W		Opens the [Locals] window.
	Code	Eventpoints	Ctrl + E		Opens the [Event] window.
		Trace	Ctrl + T		Opens the [Trace] window.
		Stack Trace	Ctrl + K		Opens the [Stack Trace] window.
	Graphic	Image…	Shift + Ctrl + G		Opens the [Image] window.
		Waveform…	Shift + Ctrl + V		Opens the [Waveform] window.

Table B.1 Window Functions (cont)

Menu	Option		Shortcut	Toolbar Button	Remarks
Setup	Radix	Hexadecimal			Uses a hexadecimal for displaying a radix in which the numerical values will be displayed and entered by default.
		Decimal			Uses a decimal for displaying a radix in which the numerical values will be displayed and entered by default.
		Octal			Uses an octal for displaying a radix in which the numerical values will be displayed and entered by default.
		Binary			Uses a binary for displaying a radix in which the numerical values will be displayed and entered by default.
	Emulator	System…			Opens the [Configuration Properties] dialog box allowing the user to modify the debugging platform settings.
		Emulator Setting…			Opens the [Emulator Setting] dialog box.
		Start/Stop Function Setting…			Opens the [Start/Stop Function Setting] dialog box.
Debug	Debug Sessions…				Opens the [Debug Sessions] dialog box to list, add, or remove the debug session.
	Debug Settings…				Opens the [Debug Settings] dialog box to set the debugging conditions or download modules.
	Reset CPU				Resets the target hardware and sets the PC to the reset vector address.
	Go		F5		Starts executing the user program at the current PC.
	Reset Go		Shift + F5		Resets the target microcomputer and executes the user program from the reset vector address.

Table B.1 Window Functions (cont)

Menu	Option	Shortcut	Toolbar Button	Remarks
Debug (cont)	Go To Cursor			Starts executing the user program at the current PC until the PC reaches the address indicated by the current text cursor position.
	Set PC To Cursor			Sets the PC to the address at the row of the text cursor.
	Run…			Launches the [Run Program] dialog box allowing the user to enter the PC or PC breakpoint during executing the user program.
	Display PC	Shift + Ctrl + Y		Displays the current PC value.
	Step In	F11		Executes a block of user program before breaking.
	Step Over	F10		Executes a block of user program before breaking. If a subroutine call is reached, then the subroutine will not be entered.
	Step Out	Shift + F11		Executes the user program to reach the end of the current function.
	Step…			Launches the [Step Program] dialog box allowing the user to modify the settings for stepping.
	Step Mode Auto			Steps only one source line when the [Source] window is active. When the [Disassembly] window is active, stepping is executed in a unit of assembly instructions.
	Assembly			Executes stepping in a unit of assembly instructions.
	Source			Steps only one source line.
	Halt Program	Esc		Stops the execution of the user program.
	Initialize			Disconnects the debugging platform and connects it again.
	Connect			Connects the debugging platform.

Table B.1 Window Functions (cont)

Menu	Option	Shortcut	Toolbar Button	Remarks
Debug (cont)	Disconnect			Disconnects the debugging platform. This option cannot be used in some products.
	Save Memory…			Saves the specified memory area data to a file.
	Verify Memory…			Verifies file contents against memory contents.
	Download Modules			Downloads the object program.
	Unload Modules			Unloads the object program.

Appendix C Command-Line Functions

The emulator supports the commands that can be used in the command-line window.

For details, refer to the online help.

Appendix D Notes on High-performance Embedded Workshop

1. Note on Moving Source File Position after Creating Load Module

   When the source file is moved after creating the load module, the [Open] dialog box may be displayed to specify the source file during the debugging of the created load module. Select the corresponding source file and click the [Open] button.

2. Source-Level Execution

   — Source file

   Do not display source files that do not correspond to the load module in the program window. For a file having the same name as the source file that corresponds to the load module, only its addresses are displayed in the program window. The file cannot be operated in the program window.

   — Step

   Even standard C libraries are executed. To return to a higher-level function, enter Step Out. In a for statement or a while statement, executing a single step does not move execution to the next line. To move to the next line, execute two steps.

3. Operation During Accessing Files

   Do not perform other operations during downloading the load module or saving in the [Verify Memory], [Save Memory], or [Trace] window because this will not allow correct file accessing to be performed.

4. Watch

   — Local variables at optimization

   Depending on the generated object code, local variables in a C source file that is compiled with the optimization option enabled will not be displayed correctly. Check the generated object code by displaying the [Disassembly] window.
   If the allocation area of the specified local variable does not exist, displays as follows.

   Example: The variable name is asc.

   asc = ? - target error 2010 (xxxx)

   — Variable name specification

   When a name other than a variable name, such as a symbol name or function name, is specified, no data is displayed.

   Example: The function name is main.

   main =

   A variable that is allocated to register cannot be selected for monitoring.

— Variable updating
For updating of the content of the variable that has been registered in the [Watch] window, read the data after the break in user program execution. The realtime operation for the user program is disabled because the user program is temporarily stopped.

5. Line Assembly
— Input radix
Regardless of the Radix setting, the default for line assembly input is decimal. Specify H' or 0x for the H8 series microcomputers and xxxxh for the R8C/Tiny, 740, M32C and M16C series microcomputers as the radix for a hexadecimal input.

6. Command Line Interface
— Batch file
To display the message "Not currently available" while executing a batch file, enter the sleep command. Adjust the sleep time length which differs according to the operating environment.

Example: To display "Not currently available" during memory_fill execution:

```
sleep d'3000

memory_fill 0 ffff 0
```

— File specification by commands
The current directory may be altered by file specifications in commands. It is recommended to use absolute paths to specify the files in a command file so that the current directory alteration is not affected.

Example:
```
FILE_LOAD C:\Hew3\Tools\Renesas\DebugComp\Platform\E8a\
E8aR8C\TutorialR8C\Tutorial\Debug_R8C_E8a_SYSTEM\Tutorial.x30
```

7. Memory Save During User Program Execution
Do not execute memory save or verifying during user program execution.

8. Load of Motorola S-type Files
This High-performance Embedded Workshop does not support Motorola S-type files with only the CR code (H'0D) at the end of each record. Load Motorola S-type files with the CR and LF codes (H'0D0A) at the end of each record.

9. [I/O] Window
— Display and modification
Do not change values in the [I/O] window because the emulator uses the address break controller.
— Note that the E8a emulator does not support the display of the invalid module in the [I/O] window. For the H8/Tiny and H8/Super Low Power series, the E8a emulator dose not support the display of the bit information in the [I/O] window.

10. Note on [Register] Window Operation During Program Execution

The register value cannot be changed in the [Register] window during program execution. Even if the changed value is displayed, the register contents are not changed actually.

11 Using Register Contents

Use the value contained in a CPU register by specifying the register name prefixed by the “#” character, e.g.: #R1, #PC, #R6L, or #ER3 when you are entering a value elsewhere in the High-performance Embedded Workshop, for example when displaying a specified address in the [Disassembly] or [Memory] window.

In the R8C/Tiny, specify and enter the bank number before a register name; e.g., #0R0 or #0R0H.

12. Break Functions

— When the PC breakpoint is set in the flash memory area, the program is written to the flash memory each time the user program is executed. At this time, note that the number of rewritable times will be decreased.

— BREAKPOINT cancellation

When the contents of the BREAKPOINT address are modified during user program execution, the following message is displayed when the user program stops.

BREAKPOINT IS DELETED A=xxxxxxxx

If the above message is displayed, cancel all BREAKPOINT settings with the [Delete All] or [Disable] button in the [Breakpoints] window.

13. Number of BREAKPOINT and [Stop At] Settings in the [Run...] Menu

The maximum number of BREAKPOINTs and [Stop At] settings allowed in the [Run...] menu is 255. Therefore, when 255 BREAKPOINTs are set, specification by [Stop At] in the [Run...] menu becomes invalid. Use the BREAKPOINTs and [Stop At] in the [Run...] menu with 255 or less total settings.

14. Note on RUN-TIME Display

The execution time of the user program displayed in the [Status] window is not a correct value since the timer in the host computer has been used.

15. Note on Displaying `Timeout error`

If `Timeout error` is displayed, the emulator cannot communicate with the target microcomputer. Turn off the emulator and the user system and connect the emulator again by using the High-performance Embedded Workshop.

16. Support of Double Float Format

In the following memory operations, the double float format is not supported:

— [Fill Memory] dialog box

— [Search Memory] dialog box

— MEMORY_FILL command

The [Format] specification in the [Copy Memory] dialog box is ignored. Memory is copied in a byte unit.

17. Note on Using the [Run Program] Dialog Box

When [Run...] is selected from the [Debug] menu to specify the stop address, there is the following note:

— When the breakpoint that has been set as Disable is specified as the stop address, note that the breakpoint becomes Enable when the user program stops.

18 [Stack Trace] window

When using the NC30WA or NC8C, if parameters and local variables of a function are referred in the call stack window and one or more parameters are on a register, the register parameters may be interchanged with the local variables. However, the correct values are displayed.

For NC308WA, NC30WA, or NC8C, specify the compiler option '-genter', which is used to output the ENTER or EXITD instruction, to all functions. If this option is not specified, the [Stack Trace] window will not be correctly displayed.

19. Memory Access during User Program Execution

When a memory is accessed from the memory window, etc. during user program execution, the user program is resumed after it has stopped in the E8a emulator to access the memory. Therefore, realtime emulation cannot be performed

20 Internal Flash Memory

The emulator is also available as the programming tool for mass production.
However, the actual MCU, which has been used by connecting to the emulator, has been programmed to the flash memory at emulation and stressed. Do not use the MCU that has been used for debugging for the mass products.

21 Suspend during Emulator Use

The host computer cannot enter suspend mode while the emulator is in use.

22 Verification of Memory

Use the FILE_VERIFY command to verify memory. Do not use [Verify Memory…] in the [Debug] menu.

23 Overwrite of a File

In command line interface, a file that has the same name as the output file is overwritten without asking the user.

24 Usage with Other Products

To install the SuperH™ RISC engine C/C++ compiler package or H8, H8/300 series C/C++ compiler package after installing the emulator debugger for High-performance Embedded Workshop, select [High-performance Embedded Workshop] as the component to be installed in [Select Components] during installation of the compiler package.

25 Download of Load Modules

When load modules are downloaded, select [Debug] -> [Download] -> [All Download modules].

26 [Program Flash] Mode

When microcomputers are continuously programmed, be sure to turn the target on or off.

27 Memory Access in the [Program Flash] Mode

Memory cannot be accessed in the [Program Flash] mode. In this mode, values displayed in the [Memory] or [IO] window are dummy.

28 Memory Access during Flash Memory Programming

During flash memory programming (e.g., user program execution), operation for memory accessing such as opening the [Memory] window is not allowed. Values displayed here are dummy. Access the memory again after flash memory programming has been completed.

Appendix E E8aSCP (E8a Self-Check Program)

This appendix describes the troubleshooting procedure that applies the self-check program for the E8a emulator (E8aSCP).

E.1. System Setup to Execute the E8aSCP

(1) The following equipment is required to execute the check program. Do not connect the user system interface cable or the user system while the check program is running.

- E8a emulator main unit
- USB cable included with the E8a emulator
- User system interface cable included with the E8a emulator
- CD included with the E8a emulator
- Host computer
  Install the High-performance Embedded Workshop and the USB driver on the host computer in advance.

(2) Open the cover of the self-check connector for the emulator. Connect the user system interface cable to the user-system connector and the self-check connector.

⚠ WARNING

Fully insert the user system interface cable in the self-check connector as shown in figure E.2.

Figure E.1 Connecting the User System Interface Cable when Using the E8aSCP

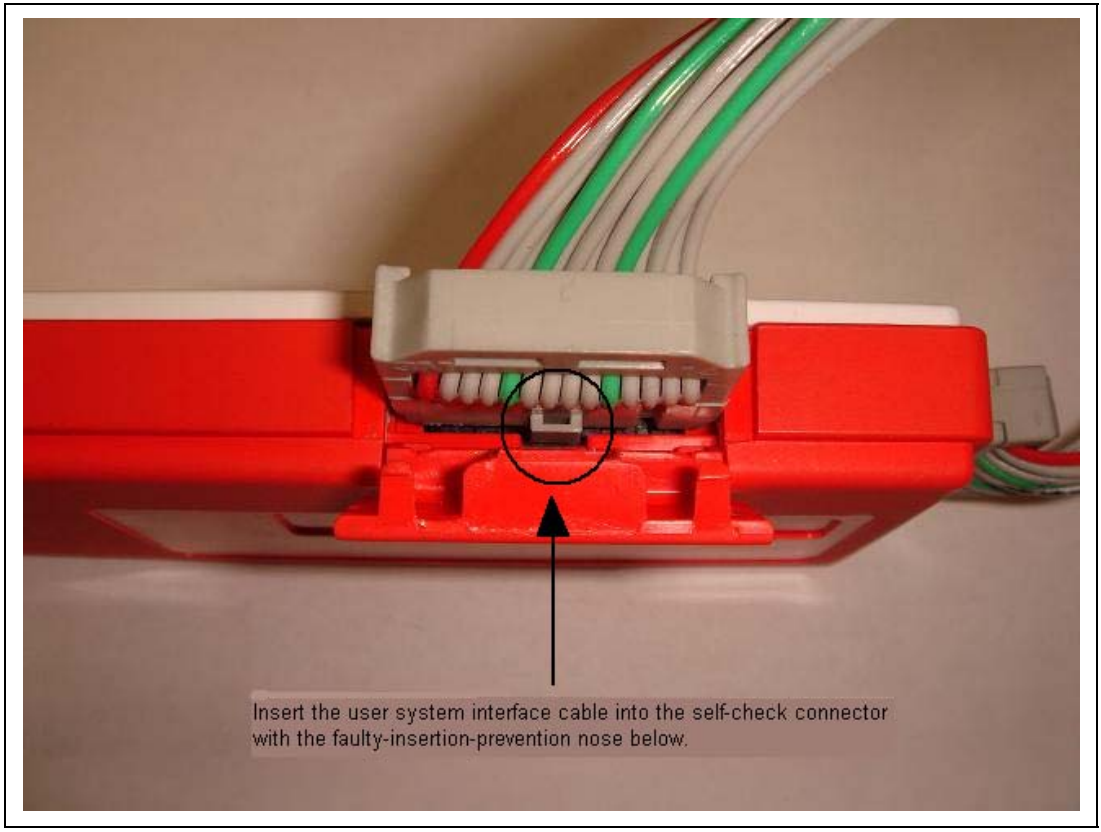


Figure E.2 Connecting the User System Interface Cable to the Self-Check Connector

(3) Start up the host computer, and connect the E8a emulator to the USB connector of the host computer via the USB cable provided.

E.2. Executing the Self-Check Program

(1) Execute [Program]-[Renesas]-[High-performance Embedded Workshop]-[E8a Self Check Program].

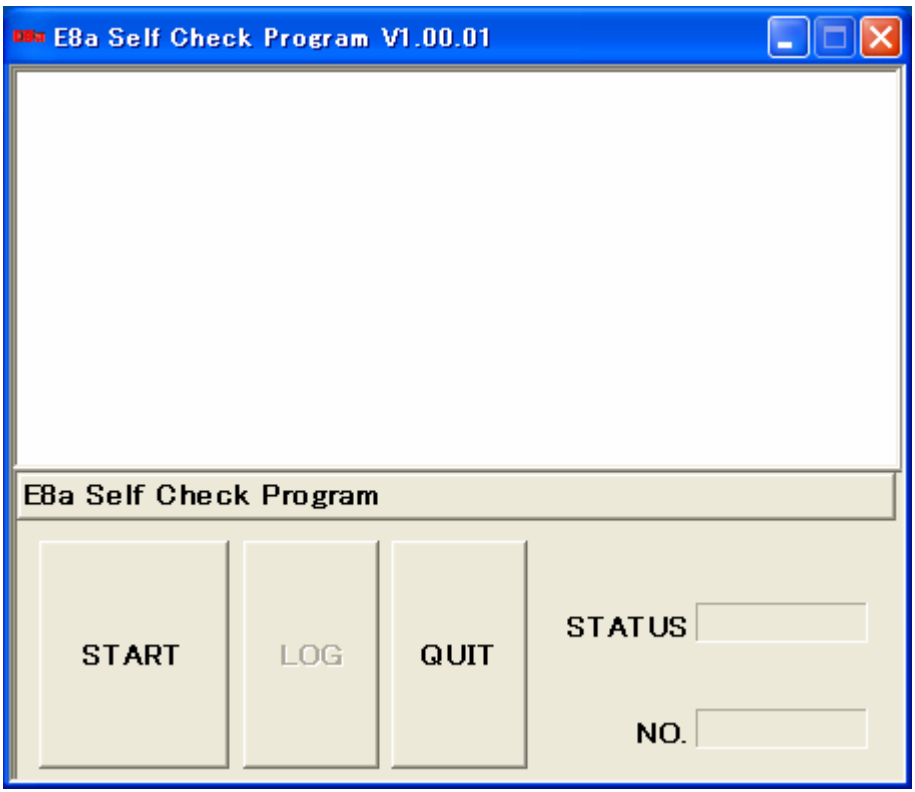


Figure E.3 Starting Up the E8aSCP

(2) Click the ‘START’ button. The following message will be displayed:
‘Please confirm that the User interface cable is connected to the self-check connector in a correct direction’.

Confirm that the user system interface cable is connected to the self-check connector in the correct direction, as shown in figure E.2, and click the [OK] button. The self-check program will start.

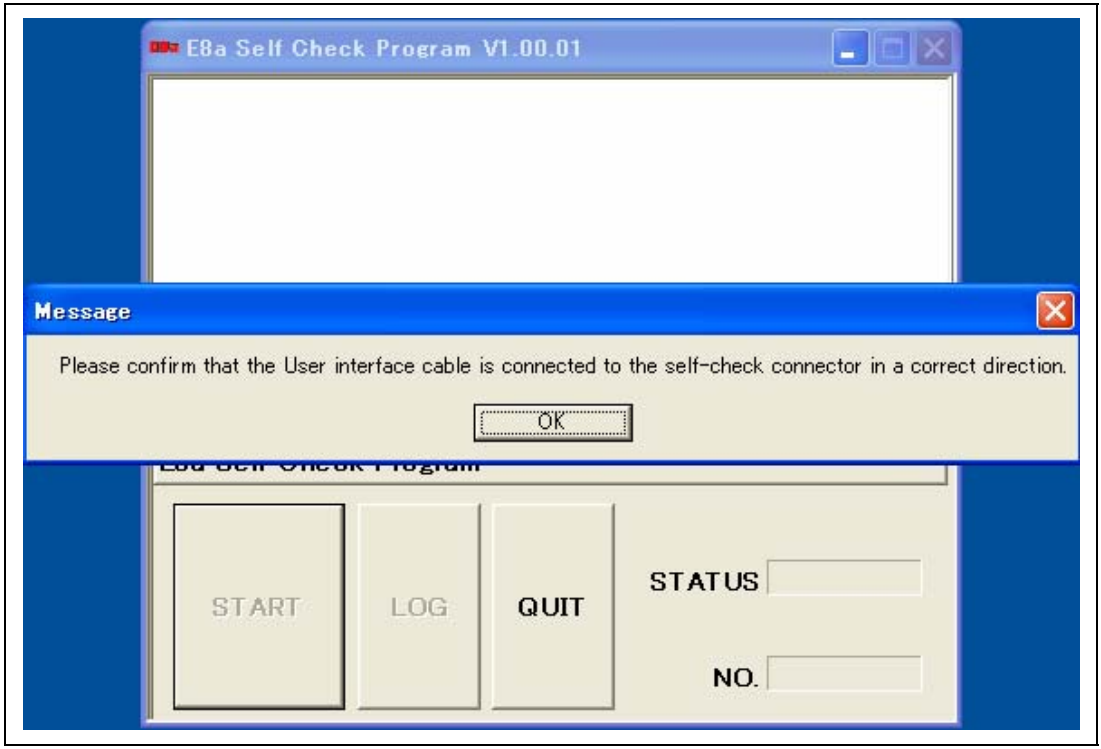


Figure E.4 Confirming Connection of the User System Interface Cable

(3) The message ‘TEST1 is running.’ or ‘TEST2 is running.’ will appear in the window. This indicates the start of the self-check program.

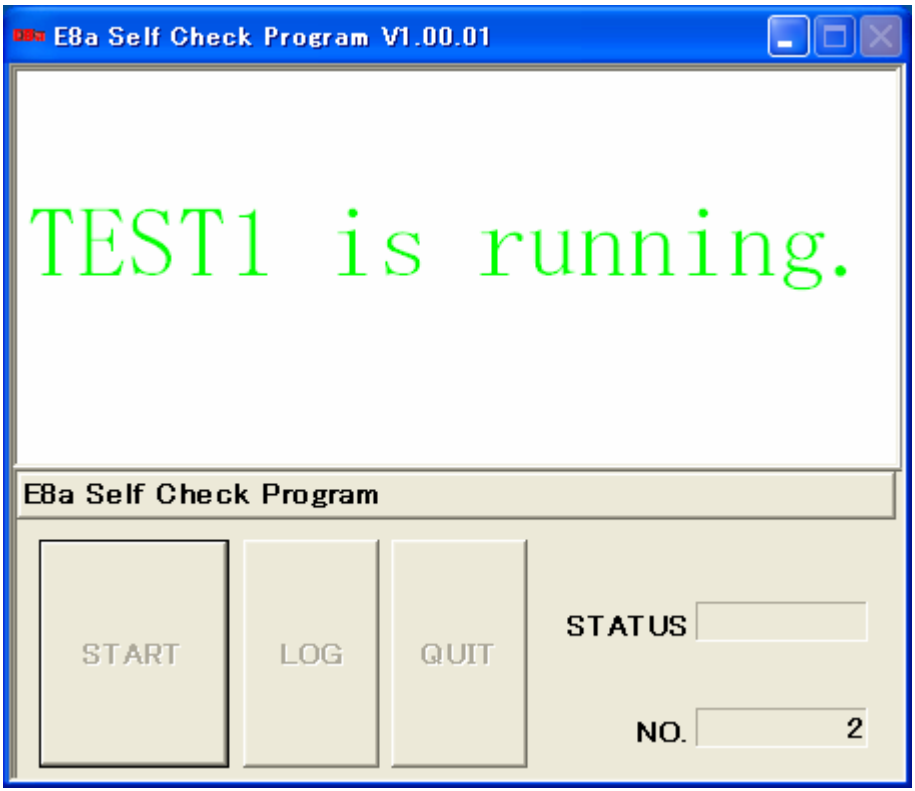


Figure E.5 Running the Self-Check Program

(4) When ‘TEST3 is waiting.’ starts, the following message will be displayed:
‘Please remove the User interface cable from the self-check connector to execute TEST3.’

When the above message is displayed, remove the user system interface cable from the self-check connector and close the cover of this connector. Click the [OK] button. The self-check program will be continuously executed.

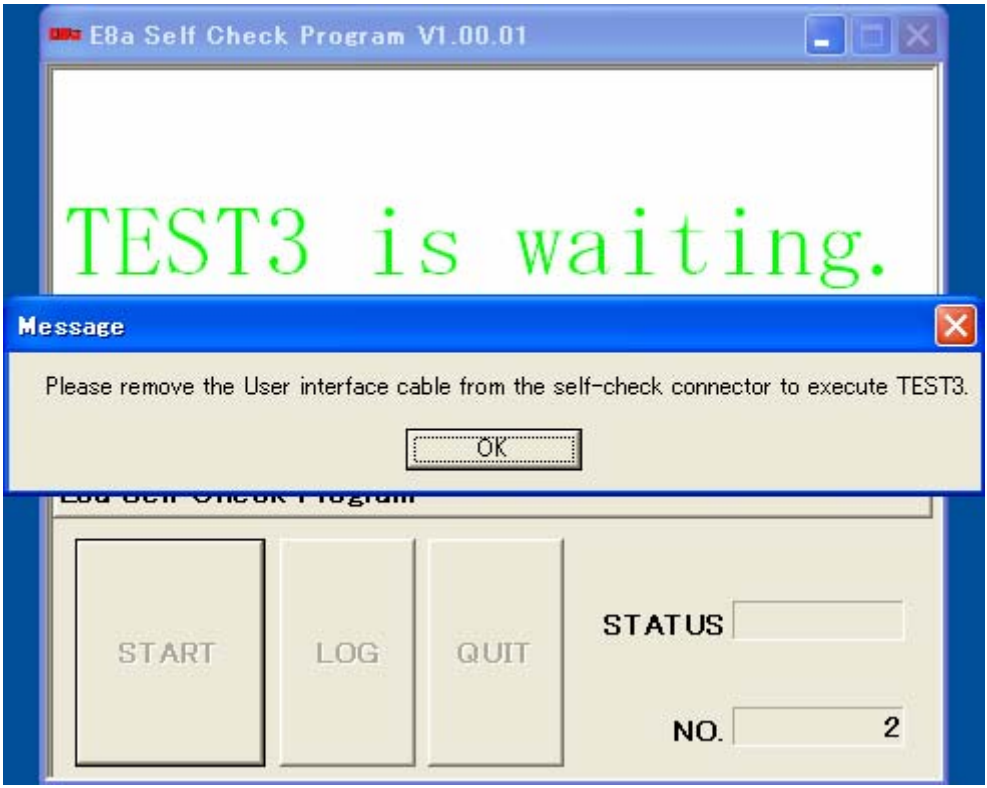


Figure E.6 ‘Please remove the User interface cable from the self-check connector to execute TEST3.’ Message

(5) The message ‘TEST3 is running.’ will appear in the window. This starts execution of the self-check program.

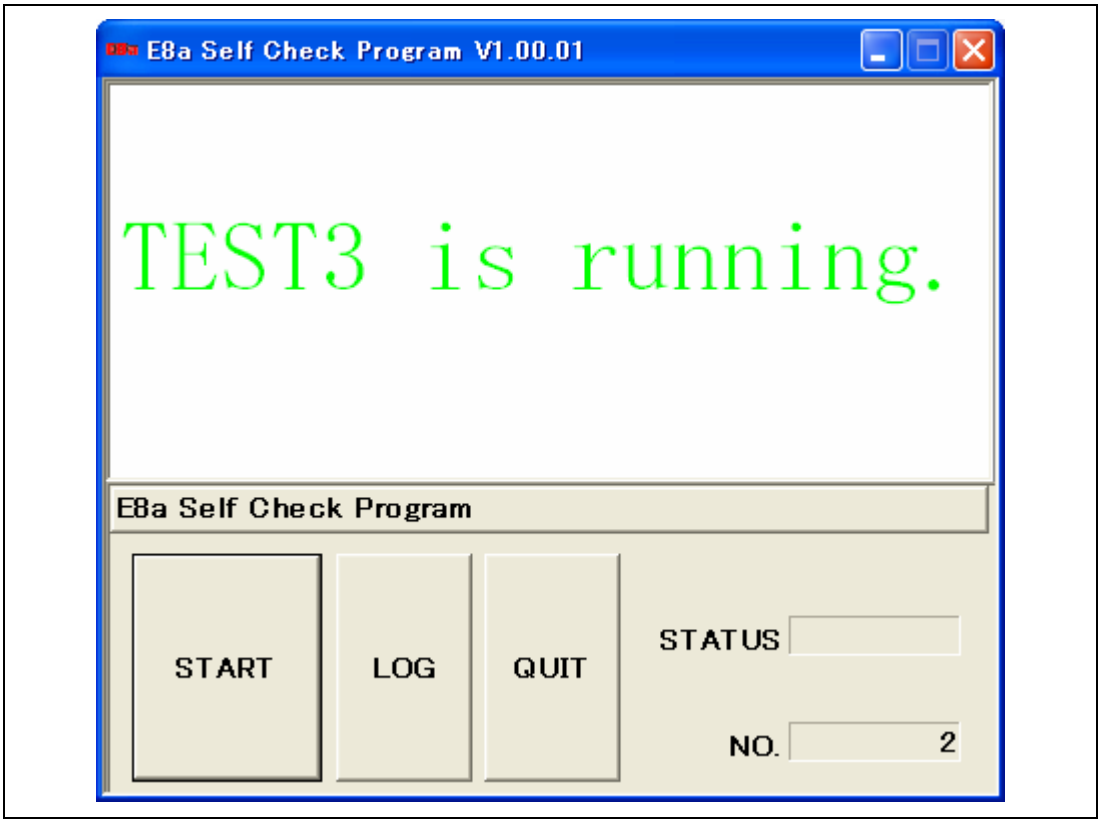


Figure E.7 Running the Self-Check Program

(6) The message ‘PASS’ or ‘FAIL’ will be displayed. ‘PASS’ indicates that the emulator is working correctly. Click the [QUIT] button to exit the self-check program.

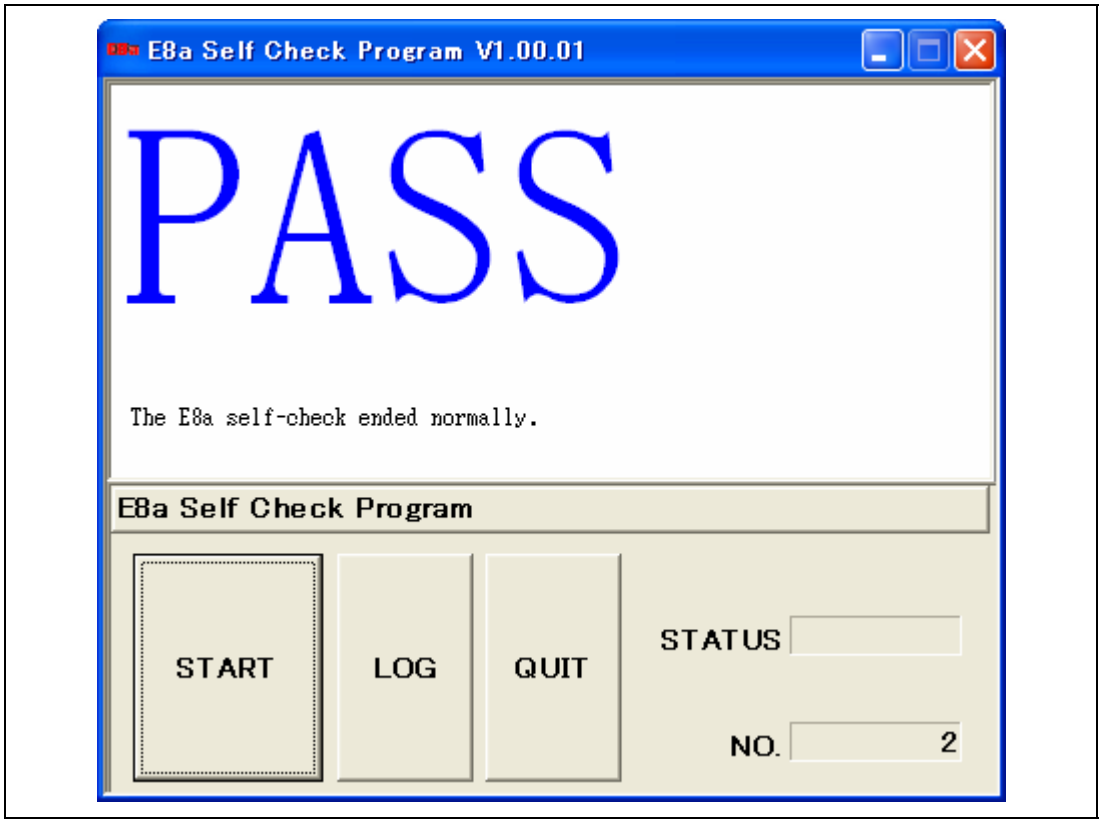


Figure E.8 Quitting the Self-Check Program

If ‘FAIL’ is displayed during execution of the self-check program, the content of an error is displayed at the center of the window. Table E.1 shows error messages.

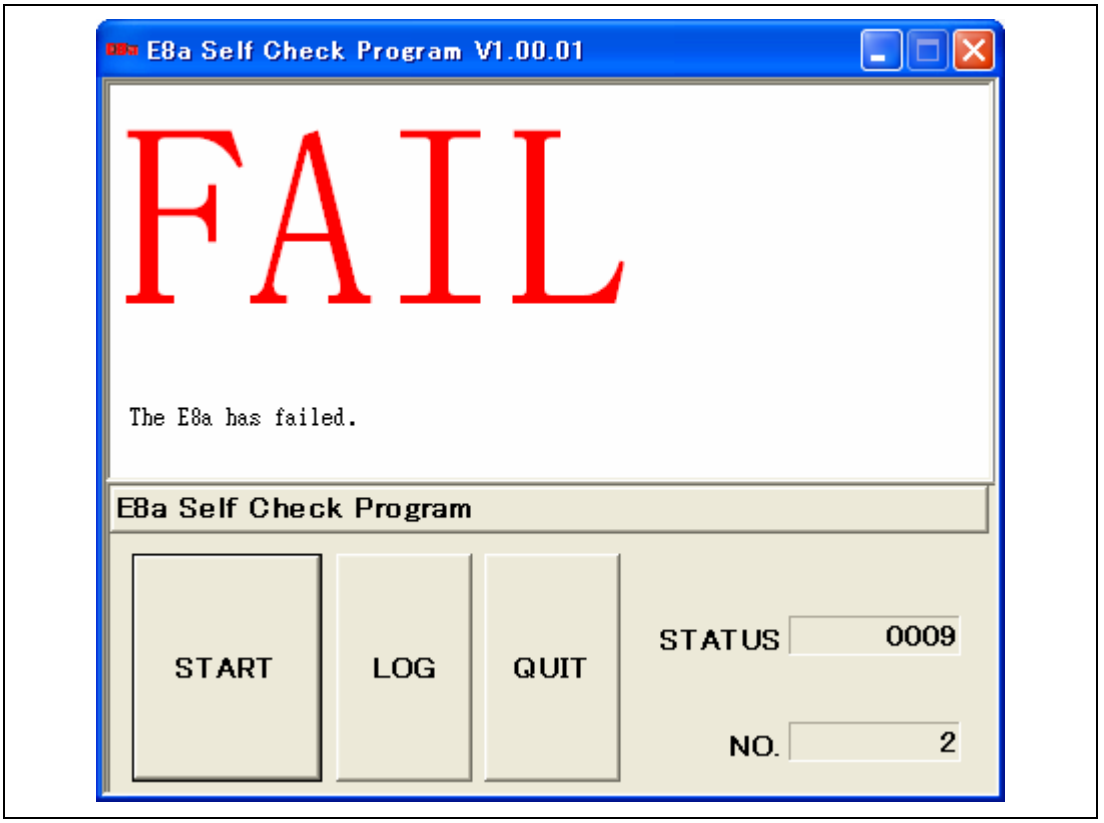


Figure E.9 Window Showing an Error

Clicking the [LOG] button opens the dialog box for selecting a file. Specify the destination to save the result of program execution. The result will be saved as a text file.

Table E.1 Error Messages

Error Message	Description
The E8a has failed.	The E8a emulator has become faulty. If this applies to the contents guaranteed in 'IMPORTANT' described in this user's manual, we will repair or replace your faulty product free of charge. If this guarantee does not apply to your product's fault, we will repair the product or replace it with a new one with extra-charge. However, replacement will be lower in cost and we recommend that you purchase a new emulator.
Please connect the User interface cable with the self-check connector in a correct direction.	Connect the user system interface cable to the self-check connector in the correct direction, as shown in figure E.2, before running the self-check program.
Because the User interface cable was connected with to self-check connector, TEST3 was interrupted. Perform the self-check again from the beginning.	Execution of TEST3 was halted because the user system interface cable was not removed from the self-check connector before starting ‘TEST3 is running.’ Connect the user system interface cable to the self-check connector in the correct direction, as shown in figure E.2, and restart the self-check program from the beginning.
An error occurred during USB I/O processing.	The USB cable was disconnected during execution of the self-check program, or the E8a emulator became faulty. Execute the self-check program again. If this error message is displayed again, the emulator could be faulty.

Renesas Microcomputer Development Environment System
User's Manual
E8a Emulator

Publication Date: 1st Edition, June 21, 2007
Published by: Sales Strategic Planning Div.
Renesas Technology Corp.
Edited by: Customer Support Department
Global Strategic Communication Div.
Renesas Solutions Corp.

© 2007. Renesas Technology Corp., All rights reserved. Printed in Japan.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

Renesas Technology America, Inc.
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

Renesas Technology Hong Kong Ltd.
7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2730-6071

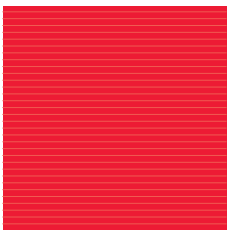
Renesas Technology Taiwan Co., Ltd.
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology Singapore Pte. Ltd.
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

E8a Emulator
User's Manual



Renesas Technology Corp.
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan